AFRL-RH-WP-TR-2013-0108

A SUBMODULARITY FRAMEWORK FOR DATA SUBSET SELECTION

Katrin Kirchhoff
Jeff Bilmes
Kai Wei
Yuzong Liu
Kai Wei
Arindam Mandal
Chris Bartels

Department of Electrical Engineering
University of Washington
Box 352500
Seattle, WA 98194

September 2013

FINAL TECHNICAL REPORT

Distribution A. Approved for public release; distribution unlimited.

AIR FORCE RESEARCH LABORATORY
711TH HUMAN PERFORMANCE WING
HUMAN EFFECTIVENESS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

Qualified requestors may obtain copies of this report from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RH-WP-TR-2013-0108 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//Signature//	//Signature//
WILLIAM KILPATRICK Work Unit Manager Human Trust and Interaction Branch	LOUISE CARTER, Ph.D. Chief, Human-Centered ISR Division Human Effectiveness Directorate 711th Human Performance Wing Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
30-SEP-2013	Final	30 July 2012 – 15 September 2013

4. TITLE AND SUBTITLE

A Submodularity Framework for Data Subset Selection

5a. CONTRACT NUMBER
FA8650-12-2-7263

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Katrin Kirchhoff Kai Wei Arindam Mandal
Jeff Bilmes Yuzong Liu Chris Bartels

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
H0CL (OMSS0005)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)

Department of Electrical Engineering
University of Washington
Box 352500
Seattle, WA 98195

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Materiel Command
Air Force Research Laboratory
711th Human Performance Wing
Human Effectiveness Directorate
Human-Centered ISR Division
Human Trust and Interaction Branch
Wright-Patterson AFB OH 45433

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S):
AFRL-RH-TR-WP-2013-0108

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution A. Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
88ABW-2013-5110, Cleared 04 December 2013

14. ABSTRACT

This report describes the outcome of the project A Submodularity Framework for Data Subset Selection. The goal of the project was to develop and evaluate novel submodular functions for the purpose of subselecting large sets of acoustic and text data. The subselected data sets were used to train acoustic models for automatic speech recognition or translation models for machine translation, respectively. The submodular selection techniques were evaluated against random data selection and the best comparable data selection technique previously reported in the literature. Our results demonstrate that submodular data selection outperforms all baseline techniques, i.e. for a fixed data subset size, submodular selection resulted in systems with better performance. Additionally, submodular selection was applied to the problem of feature selection, where it outperformed standard modular feature selection techniques.

15. SUBJECT TERMS
Submodular functions, data selection, feature selection, automatic speech recognition, machine translation, classification

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			William Kilpatrick
U	U	U	SAR	57	19b. TELEPHONE NUMBER *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

THIS PAGE INTENTIONALLY LEFT BLANK.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This research effort was supported by the Intelligence Advanced Research Projects Activity (IARPA) via the Air Force Research Laboratory (AFRL). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

1.0 SUMMARY

This report summarizes the activities and findings of the seedling project "A Submodularity Framework for Data Subset Selection." The goal of the project was to develop and evaluate novel data subset selection algorithms that are based on the framework of submodular functions, a class of mathematical functions that have the property of diminishing returns. Certain subclasses of submodular functions have theoretical optimization guarantees and scale easily to large data sets, which makes them uniquely suited to data subset selection.

The work performed falls into a theoretical and a practical part. The theoretical work involved surveying previous approaches to data subset selection in speech and language processing and analyzing them with respect to their sub-modular properties, if any. It was found that almost all previous approaches are modular in nature, with the exception of two recent approaches (that, however, were not explicitly analyzed as such by the authors). Additional theoretical work involved the development of novel sub-modular functions for summarizing speech and language data, the development of a multi-stage sub-modular function framework, and the development of multi- layered sub-modular functions based on a feature hierarchy (also termed "deep" sub-modular functions). The practical work focused on applying sub-modular data selection to the problems of automatic speech recognition (ASR) and machine translation (MT). Data subset selection algorithms were used to sub-select speech training data for acoustic models in the case of ASR, and to sub-select parallel training data for the translation model in the case of MT. In each case two different tasks were considered, one small-scale task and one large-scale task. Sub-modular data subset selection was evaluated against two baseline selection methods in each case: random data selection, and the best or most comparable selection method previously reported in the literature. On all tasks sub-modular data selection outperformed the baseline methods: for a given data subset size, systems trained on data sets selected by a sub-modular method outperformed systems trained on sets selected by the baseline methods. The advantage was particularly striking in the MT case, where data reductions of 90% were achieved while matching or even surpassing the performance of a system trained on all of the available data.

Additional work conducted in this project included applying the sub-modular framework to the related problem of feature selection. Feature selection was evaluated on three different tasks and compared against modular baseline methods for feature selection. It was shown to outperform the baseline techniques on all three tasks. In summary, the results demonstrate the unique suitability of the sub-modular framework to selection or summarization problems.

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

2.0 INTRODUCTION

2.1 The Big Data Problem

Data of many different forms (text data, sensor data, acoustic data, image data, etc.) is currently being produced at an exponentially increasing rate (see Figure 1). It is estimated by some that 2.5 quintillion bytes (2.5 billion gigabytes) of data are currently produced every day.[1]

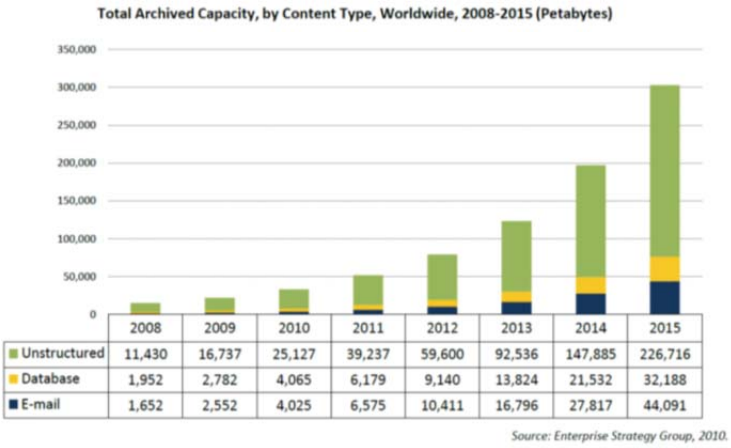

Figure 1: Amount of Data Generated World-Wide, 2008-2015

It is becoming increasingly difficult to collect and process available data for the purpose of developing machine learning based systems, such as speech recognizers, image classifiers, activity detectors, or machine translation systems. One aspect of the problem is that larger data sets require more resources, such as dedicated hardware (memory, CPUs) and processing time. However, in addition to these costs, existing software often needs to be modified or replaced to handle ever- increasing data collections, which requires considerable expertise and developer time. The biggest problem, however, is that much of the available data may be noisy, redundant with already existing/collected data, or irrelevant to the particular problem at hand. An aggravating factor in this context is that many statistical learning procedures (e.g., the Expectation-Maximization algorithm typically used for training Gaussian mixtures, or the backpropagation algorithm used for training neural networks) process training data sets repeatedly. Having unnecessary and redundant data thus results in a waste of computational resources. For these reasons it is of prime importance to develop principled ways of intelligently filtering or sub-selecting existing data collection in order to maximize the benefit to be gained from the data while minimizing cost and other resource requirements. Several data subset selection techniques have been developed previously (e.g. [1,2,3,4]); however, they are often heuristic in nature and do not provide any theoretical performance guarantees.

2.2 Data Subset Selection Scenarios

Data subset selection can be conducted in various different ways. There are a several typical scenarios that frequently occur in the fields of speech and language processing:

[1] http://www.ibm.com/big-data/us/en/

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

1. **Data Selection for Speed-Up**: Given a fixed budget (of compute cycles, financial resources, etc.), the goal is to identify a smaller subset of the data that fits the budget but provides as much information as the original large data set. This would result in significantly shorter system development time. Consider the case where a new speaker identification system needs to be developed for a new dialect within one week. 200,000 hours of in-domain speech data are available. A single development experiment for a data set of this size might take e.g.~2 days. If it was possible to select a subset that can be processed in five hours yet produces results of the same quality as the original data set, a much larger number of experiments could be completed and the resulting system would be more highly optimized.
2. **Data Selection for Adaptation**: Given a known test set, that subset of the training data is to be selected that produces the best possible performance on this test set (also called "selection for adaptation.") For example, a highly relevant set of foreign-language data has been collected that needs to be machine-translated. The goal is to fine-tune a machine translation system to this test set by sub-selecting all available training data and training the system on the resulting set, to maximize the performance of the resulting translation model.
3. **Data selection for annotation**: a set of speech data in a new language or dialect has been collected and some part of it needs to be transcribed manually to bootstrap a speech recognition system. Funding constraints limit this set to no more than three hours. Without doing any transcription, select the subset that will yield the optimum amount of information about the entire data set.
4. **Data selection for evaluation**: Given a known large test set, select a subset that is representative of the larger set such that quantitative performance measures computed on the smaller data set are proportional to the same performance measures computed on the larger set. The smaller test set may thus be used as a surrogate in order to rapidly evaluate various systems. A variant of this scenario would be to choose a subset with *balanced diversity*: The larger test may over-represent and redundantly over-evaluate certain attributes while under-evaluating others. Evaluation scores based on this test set could thus artificially be increased by tuning a system to the over-represented attributes. A subset with maximum balanced diversity would make it possible to evaluate all attributes evenly.

2.3 Project Goals and Success Criteria

In this project the first of the data subset selection scenarios described above, data selection for speed-up, was adopted for the ASR experiments. For the MT experiments the second scenario, data selection for adaptation, was adopted. The reason is that most prior work on data selection in the MT community has focused on this scenario, and all relevant baseline methods fall into this category.

The main project goal was to evaluate whether sub-modular data selection leads to better results than the comparable baselines. That is, for a fixed data set size, sub-modular data selection should result in better performance (as measured by either word error rate or BLEU) than the baseline methods. Alternatively, given a target performance number, it should be possible to reach this number using a smaller data set if sub-modular selection is used rather than either of the baseline methods. A subsidiary goal was to explore which performance level can be achieved even when reducing the data set size drastically, i.e. by one or two orders of magnitude

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

This section provides a characterization of the data subset selection problem in terms of submodular function optimization as well as a taxonomy of different submodular functions. Submodular functions [5] have been widely studied in mathematics, economics, and operations research and have more recently attracted interest in machine learning. A submodular function is defined as follows: Given a finite set of objects (samples) $V = \{v_1, ..., v_n\}$ and a function $f : 2^V \rightarrow \mathbb{R}^+$ that returns a real value for any subset $S \subseteq V$, f is submodular if $\forall A \subseteq B$, and $v \notin B$, $f(A+v) - f(A) \geq f(B+v) - f(B)$. That is, the incremental ''value'' of v decreases when the set in which v is considered grows from A to B. Powerful optimization guarantees exist for certain subtypes of submodular functions. If, for example, the function is *monotone submodular*, i.e., $\forall A \subseteq B, f(A) \leq f(B)$, then it can be maximized, under a cardinality constraint, by a greedy algorithm that scales to extremely large data sets, and finds a solution guaranteed to approximate the optimal solution to within a constant factor $1 - 1/e$ [6].

3.1 Formal Characterization of Data Subset Selection

Given a finite ground set V of size $|V| = n$ the goal is to choose the optimal subset of V as measured by some non-negative function defined on each possible subset $f : 2^V \rightarrow \mathbb{R}_+$. f can be seen as a form of ''information'' function that measures the information in a given subset $A \subset V$. Then, $f(V)$ is the value of all of the information potentially available, and the task of subset selection is to find a set A that makes $f(A)$ as large as possible. A can be considered a ''summary'' of V. Indeed, if we find an A such that $f(A) = f(V)$, then we have identified an A that (as measured by f) contains all of the information contained in V.

There are two basic optimization tasks involved in this problem. In the first case a fixed budget exists, and the goal is to find the subset $A \subset V$ that maximizes $f(A)$ but whose cost does not exceed the given budget. The budget can be quantified in a number of different ways, e.g., simply in terms of the number of items chosen. In this case the cost of A would be $|A|$, and the budget would be a number k that cannot be exceeded. The problem then becomes the following: given $1 \leq k \leq n$ find:

$$A^* \in \text{argmax} \{f(A) : |A| \leq k\} \tag{1}$$

If the function is non-decreasing (meaning supersets are never worse) then, without loss of generality, we find:

$$A^* \in \text{argmax} \{f(A) : |A| = k\} \tag{2}$$

Another way of measuring the budget is via a cost associated with each item. Let $m : V \rightarrow \mathbb{R}_+$ be a cost function such that the nonnegative cost of A is $m(A) = \sum_{a \in A} m(a)$. Given a budget specified in terms of cost, e.g. $0 < b \leq m(A)$ (without loss of generality, we may assume b is not larger than $m(A)$), the maximization problem becomes:

$$A^* \in \text{argmax} \{f(A) : m(A) \leq b\} \tag{3}$$

Such problems are called *budgeted maximization problems*.

The second kind of solution involves a quality requirement α computed by f, and the goal is to find the smallest (or cheapest) set A that satisfies a quality constraint. In this case, we are only interested

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

in subsets A that meet a quality constraint $f(A) \geq \alpha$; any subset that is of low quality is eliminated from consideration. Then, the goal is to find a set of minimum cost:

$$A^* \in \underset{S \subseteq V}{\operatorname{argmin}} |S| \text{ such that } f(S) \geq \alpha \tag{4}$$

where α is a ''cover'' requirement. A cost version of this would be

$$A^* \in \underset{S \subseteq V}{\operatorname{argmin}} m(S) \text{ such that } f(S) \geq \alpha \tag{5}$$

In general, we might take $\alpha = f(V)$ which means we are asking for the smallest set that has the same ''quality'' (or as much information as V). However, defining a new function $f'(A) = \min\{f(A), \alpha\}$, we can take any α. Hence, we have an equivalent formulation:

$$S^* \in \underset{S \subseteq V}{\operatorname{argmin}} |S| \text{ such that } f'(S) \geq f'(V) \tag{6}$$

which can be seen as a general form of this problem. We call such problems *minimum-cost covering problems*.

There are many other versions of this problem. For example, consider the following constrained maximization problem:

$$\max_{X \in \mathscr{C}} f(X)$$

where $f : 2^V \to \mathbb{R}$ is a discrete set function on subsets of a ground set $V = \{1, 2, \cdots, n\}$, and $\mathscr{C} \subseteq 2^V$ is a family of feasible solution sets. The set $\mathscr{C}$ could express, for example, that solutions must be an independent set in a matroid, a limited budget knapsack, or a cut (or spanning tree, path, or matching) in a graph. Without making any further assumptions about f, the above problems are trivially worst-case exponential time and, moreover, inapproximable.

There are other problems as well that could be considered. For example, there are ''welfare'' problems where we have K functions $f_1, f_2, \ldots, f_K : 2^V \to \mathbb{R}_+$ and the goal is to partition V into blocks $V_1, V_2, \ldots, V_k$ so that the overall ''welfare'' of the community $f_1(V_1) + f_2(V_2) + \cdots + f_K(V_K)$ is maximized.

3.2 Taxonomy of Submodular Functions

Submodular functions [5] have been widely studied in mathematics, economics, and operations research and have more recently attracted interest in machine learning. A submodular function is defined as follows: Given a finite set of objects (samples) $V = \{v_1, ..., v_n\}$ and a function $f : 2^V \to \mathbb{R}^+$ that returns a real value for any subset $S \subseteq V$, f is submodular if $\forall A \subseteq B$, and $v \notin B$, $f(A + v) - f(A) \geq f(B + v) - f(B)$. That is, the incremental ''value'' of v decreases when the set in which v is considered grows from A to B. Powerful optimization guarantees exist for certain subtypes of submodular functions. If, for example, the function is *monotone submodular*, i.e., $\forall A \subseteq B, f(A) \leq f(B)$, then it can be maximized, under a cardinality constraint, by a greedy algorithm that scales to extremely large data sets, and finds a solution guaranteed to approximate the optimal solution to within a constant factor $1 - 1/e$ [6]. As part of our theoretical work on this project a taxonomy of submodular functions relevant for the problem of data subset selection was developed. This includes primarily the various types of polymatroid functions. In the following sections we first present the basic properties of polymatroid functions before discussing the taxonomy in detail in Section 3.2.2.

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

3.2.1 Polymatroid Functions, Diversity and Fidelity, and Abstract Combinatorial Geometries.

Polymatroid functions f are normalized non-negative monotone non-decreasing submodular functions. Their formal definition is as follows:

Definition 1. *Polymatroid function*

1. $f(\emptyset) = 0$ *(normalized)*
2. $f(A) \leq f(B)$ *whenever* $A \subseteq B$ *(monotone non-decreasing)*
3. $f(A) \geq 0$ *for all* $A \subseteq V$ *(non-negative)*
4. $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$ *(submodular)*

Note that 1 and 2 implies 3. Also note that 3 and 4 implies *subadditivity*, meaning

$$f(A \cup B) \leq f(A) + f(B) \qquad \textit{(subadditivity)} \tag{7}$$

Any function that satisfies the above properties is called a *polymatroid function*. When the function is polymatroid then the problem in Equation 1 has a $1 - 1/e$ approximation guarantee [6] and the problem in Equation 4 has a $O(f(V))$ guarantee [7].
Computational benefits aside, why might a polymatroid function be an appropriate model for the data subset selection problem, and more generally, why are polymatroid functions appropriate as ''information'' functions? We explain the connection by examining what is perhaps the most famous information function, namely the entropy function from information theory.
We can view the entropy function defined on subsets of random variables in the following way. Let $V = \{1, 2, \ldots, n\}$ be an index set for n random variables $X_1, X_2, \ldots, X_n$ that is governed by a joint probability distribution $p(x_1, x_2, \ldots, x_n)$ over those n variables. We can take any subset of variable indices $A \subset V$, with $A = \{a_1, a_2, \ldots, a_k\}$ with $a_i \in \{1, \ldots, n\}$. Then we can construct a set function as follows:

$$f_H(A) = H(X_A) = H(X_{a_1}, X_{a_2}, \ldots, X_{a_k}) \tag{8}$$

$$= -\sum_{x_{a_1}, x_{a_2}, \ldots, x_{a_k}} p(x_{a_1}, x_{a_2}, \ldots, x_{a_k}) \log p(x_{a_1}, x_{a_2}, \ldots, x_{a_k}) \tag{9}$$

where we obtain the marginal as follows

$$p(x_{a_1}, x_{a_2}, \ldots, x_{a_k}) = \sum_{x_{V \setminus A}} p(x_V). \tag{10}$$

Given an $A \subset V$ and a $v \notin A$, then if X_v is independent of X_A we have $f(A + v) = f(A) + f(v)$, meaning the entropy is additive in this case --- this is the largest possible value since it always follows that $f(A + v) \leq f(A) + f(v)$. If, on the other hand, X_v is determined by X_A (complete dependence), then $f(A + v) = f(A)$, so X_v is completely redundant with A, it provides no new information. Since it is an entropy function, moreover, we always have that $f(A + v) \geq f(A)$, so monotonicity holds. Also, it is possible to express partial dependence as:

$$f(A) < f(A + v) < f(A) + f(v) \tag{11}$$

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

where we note that the inequalities are strict. In such case, v is partially dependent on A but it is neither entirely redundant nor is it entirely independent --- there is some additional information over A that v provides, but it is not the case that the information in v is entirely separate from A. Moreover, the property that ''further conditioning reduces entropy'' is expressed as follows:

$$H(X_A|X_B) = H(X_{A\cup B}) - H(X_B) \leq H(X_A) \tag{12}$$

This corresponds precisely to the subadditivity property of entropy, namely:

$$f(A\cup B) = H(X_{A\cup B}) \leq H(X_A) + H(X_B) = f(A) + f(B) \tag{13}$$

Indeed, the entropy function is submodular, and this follows by the non-negativity of conditional mutual information. Indeed given any three disjoint sets A, B, C, we have

$$I(X_A; X_B|X_C) = H(X_A, X_C) + H(X_B, X_C) - H(X_A, X_B, X_C) - H(X_C) \geq 0 \tag{14}$$

Thus, with an entropy function, we can measure complete independence, complete dependence, and partial independence only via the inequalities on entropy.
As it turns out, any polymatroid function possesses all of the properties of the entropy function that make the entropy function a natural measure of information. Hence, a polymatroid function can also be seen as an information function, in what is sometimes known as an ''abstract independence'' space, or in a ''combinatorial geometry.'' This space is not necessarily Euclidean (although it can be, see the matrix rank example below).
Firstly, for any polymatroid function f, define its conditional valuation: for any $A, B \subseteq V$

$$f(A|B) \triangleq f(A\cup B) - f(B). \tag{15}$$

This is known as the gain of A in the context of B. Because of submodularity, we have that $f(A|B) \leq f(A|B')$ for any $B \subseteq B'$ (which in fact is the definition of submodularity). Hence, such a construct, which we could call 'conditioning reduces valuation'', defines the very nature of submodular functions itself. When we let f represent an information function, generalizing the entropy function, we can immediately define a set of functions analogous to those which are defined for entropy. We list them all here.
Given an arbitrary polymatroid function f, we obtain the following natural definitions and properties (all of which are parameterized by the function f):

Definition 2. *Information of a set. The information of a set A is given by the polymatroid valuation of that set $f(A)$.*

Definition 3. *Conditional information of a set. Conditional information of a set A given B is given by $f(A|B) = f(A\cup B) - f(B)$.*

Definition 4. *Abstract mutual information. The abstract mutual information between two sets A and B is given by $I_f(A;B) = f(A) + f(B) - f(A\cup B)$*

Definition 5. *Abstract conditional mutual information. Abstract conditional mutual information between two sets A and B given C is given by*
$I_f(A;B|C) = f(A\cup C) + f(B\cup C) - f(A\cup B\cup C) - f(C)$

Distribution A. Approved for public release; distribution unlimited.
88 ABW-2013-5110; Cleared 04 December 2013

Definition 6. *Generalized abstract independence. Sets A and B are independent if* $I_f(A;B) = 0$

Two sets are not independent if $I_f(A;B) > 0$ meaning that $f(A) + f(B) > f(A \cup B)$, or subadditivity holds at A and B, meaning there is some redundancy, as measured by f, between A and B. A is completely redundant with B if it is the case that $f(A|B) = 0$ meaning $f(A \cup B) = f(B)$. This means that A brings no additional valuation when considered in the context of B.

Definition 7. *Generalized abstract conditional independence. Sets A and B are independent given C if* $I_f(A;B|C) = 0$

Property 8. *Mutual information is non-negative. We have* $I_f(A;B|C) \geq 0$ *which follows from submodularity of* f.

Property 9. *Conditioning reduces valuation.* $f(A|B) \geq f(A|B')$ *for any A and* $B \subseteq B'$.

Hence, it is reasonable, at least from a representational point of view, to use a polymatroid function as a model of information. Why might we wish to use a polymatroid function rather than something well known such as entropy? After all, entropy has been well studied and widely used not just in communications theory but in many other fields (including statistics and machine learning, and others as diverse as sociology and neuroscience). There are several reasons for this:

- Entropy requires the elements V to be thought of as random variables.
- Entropy requires a probability distribution over those random variables. This distribution needs to be obtained from somewhere, typically learned via some training data set or specified by a scientist.
- Each query $f(A)$ requires probabilistic inference, something that if done exactly has time and memory costs exponential in $|A|$. Hence even a single entropy query can be computationally intractable.
- While entropy functions are very diverse, the space of entropy functions does not include all polymatroid functions. Indeed, there is an additional inequality that entropy must satisfy that polymatroid functions need not satisfy.

3.2.2 Types of Polymatroid Functions and their Properties.

In this section we provide a more detailed taxonomy of polymatroid functions, their properties, and how they might be useful for the data subset selection problem. This includes the functions actually used for the experiments described in later sections of this report, but it also shows that the class of polymatroid functions is much richer and includes many functions not investigated as part of this project.

Matrix Rank: Our simplest example is the matrix rank function. Given a $m \times n$ matrix (n m-dimensional column vectors), consider the index set $[n] = \{1, 2, \ldots, n\}$ and consider $A \subseteq [n]$ to be any subset of column vector indices. Let $r(A)$ be the rank of the submatrix spanned by the column vectors indexed by the set A, so that $r(V)$ is the rank of the matrix. Then $r : 2^{[n]} \to \mathbb{Z}_+$ is a polymatroid function. This is known more generally as an instance of a matroid rank function (see below).

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Matroid Rank: To define a matroid rank function, we need a few definitions, including the definition of a matroid.

Definition 10 (Set system)**.** *A (finite) ground set V and a set of subsets of V, $\emptyset \neq \mathscr{I} \subseteq 2^V$ is called a set system, notated $(V, \mathscr{I})$.*

We note that there are in fact may ways to define a matroid, and we give only one of them here.

Definition 11 (Matroid)**.** *A set system $(V, \mathscr{I})$ is a Matroid if*

(I1) $\emptyset \in \mathscr{I}$

(I2) $\forall I \in \mathscr{I}, J \subset I \Rightarrow J \in \mathscr{I}$

(I3) $\forall I, J \in \mathscr{I}$*, with* $|I| = |J| + 1$*, then there exists* $x \in I \setminus J$ *such that* $J \cup \{x\} \in \mathscr{I}$.

A matroid rank function is defined in terms of a matroid, and corresponds to the largest independent subset of a given set.

Definition 12 (Matroid rank function)**.** *The rank of a matroid is a function $r : 2^V \to \mathbb{Z}_+$ defined by*

$$r(A) = \max\{|X| : X \subseteq A, X \in \mathscr{I}\} \tag{16}$$

We can see that if the set $\mathscr{I}$ corresponds to all sets of linearly independent column vectors of a given matrix, then the matroid rank function corresponds precisely to a matrix rank function (it is the dimensionality of the space spanned by the set of vectors indexed by the set A).
Hence, we see that matroid (and in fact any matroid) rank function measures independence (namely, $r(A+v) = r(A) + r(v) = r(A) + 1$, which means that the vector indexed by v is not in the space spanned by the vectors indexed by A), and complete dependence (namely $r(A+v) = r(A)$, and v is within the space spanned by A). What a matroid rank functions can *not* model is partial dependence as can either the entropy function or some other polymatroid function.
There is a rich history of matroids, and there are many types that go beyond the matrix-based matroids mentioned above. Even though matroids have an ''all or nothing'' approach to representing dependence they are very powerful. For example, much of the complex dependency structure of a polymatroid can be captured by a matroid rank function.

Weighted Matroid Rank: We can generalize matroid rank functions further via a modular function $m : V \to \mathbb{R}_+$. The weighted matroid rank function is given as:

Definition 13 (Weighted matroid rank function)**.** *The given non-negative modular function $m : V \to \mathbb{R}_+$, the weighted rank of a matroid is a function $f : 2^V \to \mathbb{Z}_+$ defined by*

$$f(A) = \max\{m(X) : X \subseteq A, X \in \mathscr{I}\} \tag{17}$$

This function is submodular and very general; in fact, it offers an easy way to prove submodularity of some of the functions we discuss further below.

Grouped Matroid Rank: Suppose we cluster the ground elements into not-necessarily disjoint cluster V_i as follows: $V = V_1 \cup V_2 \cup \cdots \cup V_k$. We can define a function $f : 2^{[k]} \to \mathbb{Z}_+$ as follows:

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Definition 14 (Grouped Matroid Rank)**.** *Given a matroid and a grouping as described above, for any $A \subseteq [k]$, define*

$$f(A) = r(\cup_{i \in A} V_i) \tag{18}$$

In other words, the grouping defines the blocks through which the matroid rank function is viewable. With this construct we can easily represent partial independence (since two clusters can have some overlap, and hence can have some redundancy). In fact, any rational valued polymatroid function can be exactly represented (or well approximated) in this way.
One of the issues of matroid rank functions (and the weighted and/or grouped variant) is representation. The set $\mathscr{I}$ might be exponentially big. How can we then represent a matroid and matroid rank function r in a way that is easy to produce and query values from?

Graphic Matroid Rank: Many matroids can be represented as a graphic matroid. Here, we are given a graph, and V corresponds to the set of graph edges. The matroid rank function then is, for any subset of edges A, $r(A)$ is the size of the maximum (potentially weighted) spanning tree in the subgraph induced by A. Thus, each query $r(A)$ can be computed by a maximum spanning tree algorithm which is reasonably efficient and, moreover, it is possible to represent the entire matroid as a graph.

Set Cover: Let V be our ground set and U a set that is clustered, where the clusters are indexed by elements of V, with $\{U_v\}_{v \in V}$ and $\cup_{v \in V} U_v = U$. The set cover function is defined, for $A \subseteq V$, as:

$$f(A) = |\cup_{v \in A} U_v| \tag{19}$$

which is the number of elements of U that are ''covered'' by sets indexed by elements within A. This function is polymatroid and has been widely used in computer science theory problems. Indeed cover problems are often the root of the optimization problems mentioned in Section 3.1.

Entropic Functions: We have already noted above that the widely-used entropy function from information theory is a polymatroid function. Note that since Gaussian entropy is essentially a log-determinant, this means that the log determinant of a submatrix of a positive definite matrix whose rows and columns are chosen by a set A is submodular (although it is not monotone). There are many other spectral functions of matrices as well that are polymatroidal or submodular, and that are given in [8].

Graph-Cut and Saturated Graph Cut: Given a graph $G = (V, E)$ the graph cut function is defined as follows: for any $A \subseteq V$, $f(A) = |\{(u,v) \in E : u \in A, v \in V \setminus A\}|$. A weighted variant uses an edge-weighted graph $G = (V, E, w)$ with $w : E \to \mathbb{R}_+$ and is defined as
$f(A) = w(\{(u,v) \in E : u \in A, v \in V \setminus A\})$. There are many ways to obtain the weights (for example, $w(u,v)$ could be based on a non-negative similarity score between items u and v, which is what has been used in this project). One issue with this function, however, is that it is not monotone. We can obtain a monotone variant of this function by simply using $f(S) = \sum_{v \in V, s \in S} \mathbf{1}_{\{(v,s) \in E(G)\}}$, which is the count of any edges within S or between S and $V \setminus S$, so that
$\delta(S) = f(S) + f(V \setminus S) - f(V)$ is the standard graph cut.
Neither of these functions is necessarily ideal for diversity or fidelity since they have the property that an item not chosen in the set A might be over-represented. That is, if there is a well connected element $v \notin A$ that is highly connected to many elements in A, it will increase the valuation of $f(A)$

Distribution A. Approved for public release; distribution unlimited.
88 ABW-2013-5110; Cleared 04 December 2013

extraneously. A way around this is to force the contribution of any element to saturate at a particular point. For any $i \in V$, let $C_i(A)$ be degree that A represents (or is similar to) item i. A typical instance is $C_i(A) = \sum_{a \in A} w_{i,a}$ where $w_{i,a}$ is the edge weight in the graph between vertices i and a. We can then define a *saturated graph cut* function:

$$f(A) = \sum_{i \in V} \min \{C_i(A), \alpha C_i(V)\} \tag{20}$$

where $0 < \alpha \leq 1$.
Note that the meaning of this function greatly depends on the graph, which might be e.g., a k-NN or an ε approximation graph, or it might represent a manifold of set V. The saturation coefficient α ensures that no single item i can be over-covered since once $C_i(A)$ reaches a threshold, any additions to A that would increase $C_i(A)$ has no effect (thanks to the min function) and to improve the function f overall, one must add items to A that increase $C_j(A)$ for some $j \neq i$.

Maximization and Facility Location: Given a modular vector $w \in \mathbb{R}^V_{++}$ of positive weights, $v \in V$ the function:

$$f(A) = \max_{a \in A} w_a \tag{21}$$

is polymatroidal. It is the basis for the classic *facility location function*, which has been used in operations research for many years [9].
The facility location function is another polymatroid function that is based on a weighted graph $G = (V, E, w)$ where $w : E \to \mathbb{R}_+$, although it is often best to think of this function on a complete graph (any missing edge in the graph corresponds to $w((u,v)) = 0$ for some pair $u, v \in V$). The facility location function takes the form:

$$f(A) = \sum_{v \in V} \max_{a \in A} w(v,a) p \tag{22}$$

This function is also polymatroidal since it is the sum of $|V|$ *max* functions, each of which are polymatroidal (and sums of submodular functions are submodular).
Like the graph functions mentioned in the previous section, these functions rely on each pair of objects u, v being represented by a similarity score $w(u,v)$. Any non-negative similar score can be used without danger of damaging the polymatroidal property, so this function (and in fact the graph functions of the previous section) are very flexible.
A generalization of this function relies on a softening of the *max* function. We have the following theorem:

Theorem 15.

$$\max_v w_v = \lim_{\alpha \to \infty} \frac{1}{\gamma} \log(\sum_v \exp(\gamma w_v)) \tag{23}$$

Proof. Let us assume that there are K distinct values of the vector w and let us partition the vector w into K blocks where within each block we have the same value, so $V = V_1 \cup V_2 \cup \cdots \cup V_K$ where

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

for $u, v \in V_k$ we have $w_u = w_v$. Let w_{k^*} be the maximum value of the vector. Then we have

$$\frac{1}{\gamma}\log(\sum_v \exp(\gamma w_v)) = \frac{1}{\gamma}\log(\sum_{k=1}^{K} |V_k| \exp(\gamma w_k)) \tag{24}$$

$$= \frac{1}{\gamma}\log\left(\frac{\exp(\gamma w_{k^*})}{\exp(\gamma w_{k^*})}\sum_{k=1}^{K} |V_k| \exp(\gamma w_k)\right) \tag{25}$$

$$= \frac{1}{\gamma}\log(\exp(\gamma w_{k^*})) + \frac{1}{\gamma}\log\left(\sum_{k=1}^{K} |V_k| \exp(\gamma(w_k - w_{k^*})\right) \tag{26}$$

$$= w_{k^*} + \frac{1}{\gamma}\log(|V_{k^*}| + \sum_{k \neq k^*} o(1)) \tag{27}$$

Thus,

$$\lim_{\gamma \to \infty} \frac{1}{\gamma}\log(\sum_v \exp(\gamma w_v)) = w_{k^*} \tag{28}$$

□

Given the above softmax representation, we can generalize facility location to a ''soft facility location'' function, namely:

$$f(S) = \sum_{i \in V} \frac{1}{\gamma}\log(1 + \sum_{j \in S} \exp(\gamma w_{ij})). \tag{29}$$

This function is submodular since it is a sum of log over modular functions (see below), and it becomes more like a crisp *max* function as γ gets larger. Another generalization of the *max* function is the $k - max$ function. That is, given a non-negative weight vector $w : V \to \mathbb{R}_+$, we define the $k - max$ function as:

$$f(A) = w(\{a_1^*, a_2^*, \ldots, a_k^*\}) \tag{30}$$

where

$$a_1^* \in \operatorname*{argmax}_{v \in A} w(v) \tag{31}$$

$$a_2^* \in \operatorname*{argmax}_{v \in A \setminus \{a_1^*\}} w(v) \tag{32}$$

...

$$a_k^* \in \operatorname*{argmax}_{v \in A \setminus \{a_1^*, a_2^*, \ldots, a_{k-1}^*\}} w(v) \tag{33}$$

where $|A| \geq k$. Thus, the *max* function is just the $1 - max$ function. We can easily see that this is a polymatroid function by using the weighted matroid rank function, with weight w and a k-uniform

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

matroid (all sets of size up to and including k are independent). Alternatively, plugging this function into the formula $f(A)+f(B) \geq f(A \cup B)+f(A \cap B)$ verifies submodularity.
Lastly, we note that negative *min* is submodular. Let us say that $m(v)$ is non-negative modular, then the function: $f(A) = -\min_{a \in A} m(a)$ is submodular.

Concave Over Modular: Given any modular function $m : V \rightarrow \mathbb{R}_+$ and any concave function $g : \mathbb{R}_+ \rightarrow \mathbb{R}$, we have that the function defined as

$$f(A) = g(m(A)) \tag{34}$$

is submodular. Moreover, if g is also non-decreasing then f is polymatroidal.
There are many possible g's including n^{th} root (the square root being particularly popular), or other concave functions. Some possibilities include $g(x) = \log(1+x)$ (which does not saturate), the positive side of logistic function $g(x) = \frac{1}{1+\exp(-x)}$ (which saturates), and the positive side of *tanh* (which also saturates). Another saturating concave function is $g(x) = 1 - \log(1+e^{-x})$ for $x \geq 0$. This latter saturating function and also the logistic function were the functions considered in [10].
Another useful type of concave function is to use piecewise linear functions --- these are linear functions that are stitched together at threshold points to create a concave functions. In order to guarantee submodularity one must ensure that, for a given threshold point x_i, the slope $\frac{\partial g}{\partial x}(x_i^-)$ on the left side is larger than on the right side $\frac{\partial g}{\partial x}(x_i^+)$. Sometimes these functions are particularly easy to optimize.
Another useful variant is to use a function that is initially linear followed by a concave part. When used as $f(A) = g(m(A))$, this yields a function that is modular up to some critical point before diminishing returns sets in. This can be used to ensure that a sufficient quantity of some aspect of the ground elements is reached before any partial credit is assigned. In fact, functions we saw above, such as $\min(C_i(A), \alpha C_i(V))$ fall into this category, since the function $g(x) = \min(x, \alpha)$ for some constant α is a concave function that is initially linear.

Complement Product: Suppose we have a set of weights $w : V \rightarrow [1, \infty)$ (meaning each $w(v) \geq 1$). Then it is well known that the following is submodular and decreasing.

$$f(A) = -\prod_{a \in A} w(a) \tag{35}$$

It is submodular and increasing if $0 \leq w \leq 1$, with max value $f(V) = -\prod_{v \in V} w(v)$. Hence, we can construct a polymatroid function as

$$f'(A) = \prod_{v \in V} w(v) - \prod_{a \in A} w(a) = \prod_{v \in V} w(v) - \prod_{a \in A} w(a) \tag{36}$$

One easy way to see that such functions are submodular is the following. Let $w(v) \geq 0$. Then

$$f(A) = \prod_{v \in A} w(v) = \exp \log \prod_{v \in A} w(v) = \exp\left(\sum_{v \in A} \log w(v)\right) \tag{37}$$

which is a concave over modular, and it is also supermodular, so $-f(A)$ is submodular.

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Sums of Concave over Modular: A large class of functions can be obtained in the following way. Suppose that $f_i : 2^V \rightarrow \mathbb{R}_+$ is a concave over modular function (see above), and let $\{f_i\}_{i=1}^M$ be a set of M such functions. Then we define the class:

$$f(A) = \sum_i \alpha_i f_i(A) \tag{38}$$

to be the class of sums of concave over modular functions. These have been known as ''decomposable'' submodular functions in the past [11] although we prefer not to use this terminology owing to the confusion with decomposable graphical models (which can apply to submodular functions as well in a very different way than sums of concave over modular). Hence, we will just call them SCCM functions. SCCM functions are polymatroidal as long as the mixture coefficients are non-negative and the concave functions are monotone non-decreasing. There are many types of SCCM functions as we see next.

Clustering-Based Functions: Suppose we cluster the ground elements $V = V_1 \cup V_2 \cup \cdots \cup V_k$ into k not necessarily disjoint sets. The clustering might be such that elements within the same cluster are similar, while those between two clusters are more different than similar. We can then define a polymatroid function

$$f(A) = \sum_{i=1}^{K} g(m(A \cap V_i)) \tag{39}$$

where m is a non-negative modular function, g is concave, and $V_1, V_2, ..., V_k$ are the clusters of the function. These have been used as diversity functions in the past [12, 13]. They assign diminishing returns for the repeated choice of items from the same cluster. In order to maximize f, it is necessary to choose a small number of items from each cluster, i.e., the overall set of items chosen must be diverse with repect to cluster representation. Any underrepresented minority cluster will become the most valuable at some point since the marginal gain of elements in larger communities will ultimately diminish, potentially to zero.

Bipartite Neighborhood Functions: Given a weighted bipartite graph $G = (V, U, E, w)$ where V are the left vertices, U are the right vertices, $E \subseteq V \times U$ are the edges, and $w : U \rightarrow \mathbb{R}$ is a weight function on U, it is possible to define a bipartite neighborhood function as follows. First, define the neighbor function $\Gamma_V : 2^V \rightarrow 2^U$ as we have $\Gamma_V(A) = \{u \in U : (v,u) \in E \text{ for some } v \in A\}$. Then define the function

$$f(A) = w(\Gamma(A)) \tag{40}$$

It might not seem obvious that this is a sum of concave over modular function, but indeed it is. To see this, define the neighbor function defined on subsets of U, namely $\Gamma_U : 2^U \rightarrow 2^V$ as we have $\Gamma_U(E) = \{v \in V : (u,v) \in E \text{ for some } u \in E\}$. Note also that for any $u \in U$ we have that the function

$$f_u(A) = w(u) \min(|\Gamma_U(u) \cap A|, 1) \tag{41}$$

is a concave over modular function, the reason being that $|\Gamma_U(u) \cap A|$ is modular in A (any element of V that intersects $\Gamma_U(u)$ has value 1, and other elements have value 0). Then consider the

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

function:

$$f(A) = \sum_{u \in U} w(u) \min(|\Gamma_U(u) \cap A|, 1) \tag{42}$$

This is exactly the same as above, that is we have:

$$f(A) = w(\Gamma(A)) = \sum_{u \in U} w(u) \min(|\Gamma_U(u) \cap A|, 1) \tag{43}$$

We note that it also generalizes the set cover function (we have $U_v = \Gamma_V(v)$). We can moreover obtain graph cut functions using a construct $w(\Gamma(A)) + w(\Gamma(V \setminus A)) - w(\Gamma(V))$ where each $u \in U$ has a degree of exactly two and corresponds to the edges in a standard graph.
We note that the set U can be seen as either categories, classes, concepts, or features of the ground elements V. This leads us to the next class of submodular functions.

Feature-Based Functions: Feature-based submodular functions are an important class of functions that are really just a variant of what we have already seen --- the main difference between feature functions and the bipartite neighborhood functions of the previous section is that the weight is on the edges of the bipartite graph rather than just the right-hand-side vertices. Moreover, what make this a distinct section in this document is that we can think of them in the following way that is very useful for applications: Suppose that each ground element $v \in V$ has a length-M vector $c(v) \in \mathbb{R}_+^M$ of non-negative features scores. Alternatively, if U is the set of features, let $c_u(v)$ to be the value or quantity of feature u in element v.
An example follows: Let V be a set of sentences. For $s \in \mathscr{V}$ and $u \in \mathscr{U}$, define $c_u(s)$ to be the count of feature u in sentence s. This is the number of times that feature u occurs in s. So if the sentence is:

$$s = \text{Whenever I go to New York City, I visit the New York City museum.} \tag{44}$$

So $c_{\text{the}}(s) = 1$ while $c_{\text{New York City}}(s) = 2$.
We can instantiate submodular functions of the following general form: For $A \subseteq \mathscr{V}$,

$$f(A) = \sum_{u \in U} \sqrt{\sum_{a \in A} m_f(a)} \tag{45}$$

where U is some feature set, and $m_u(a) \geq 0$ is some non-negative score that measures the degree of feature u in sentence a in some way. Rather than square root, moreover, we could use any non-decreasing concave function. Indeed, we immediately see that this is an instance of a sum of concave over modular functions. Moreover, this generalizes the bipartite graph formulation above, except that now each edge in the bipartite graph (corresponding to the matrix $\{m_u(s)\}_{u,s}$) can have its own weight.
Such functions are very useful for the instantiation of polymatroid functions in speech and natural language processing (NLP) since the community is very much used to describing objects as a set of features. For example, in NLP a sentence can be described as n-gram features, or as certain parse features. In speech, one can describe a speech utterance using a variety of acoustic or prosodic features. Feature-based submodular functions, therefore, have the ability to leverage all of the important work on both knowledge- and data-driven feature engineering that has occurred in these two communities.

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

There are, moreover, many different ways to produce the scores $m_u(a)$ other than count. For example, they can be term-frequency-inverse document frequency (TF-IDF) based normalized counts, or they can be scaled in various ways depending on the nature of the feature u. In other parts of this document, we describe how this is done.

Sums of Concave over Modular vs. Matroid Rank: Sums of concave over modular functions are indeed general, but they are not as general as matroid rank. In fact, the class of matroid rank functions and sums of concave over modular have an intersection and a symmetric difference. Even the simple matroid rank function over the four-clique graph is not possible to represent due to its inability to communicate in a very structured way.

Deep Submodular Functions: The functions described above have occurred in one form or another in the literature. The functions we describe in this section are entirely new and were developed as part of this project.
One issue with the feature-based functions described in above is that they represent interactions between different items in the ground set, but they do not represent interactions between different features or sets of features. In general, however, we wish to represent two forms of interactions:

- The interaction between sentences/items/ground-elements due to the same features being shared by those ground elements. The appropriate way to represent these interactions is:

$$f(A) = \sum_{u \in U} \text{concave}_u(\sum_{a \in A} m_u(a)) \tag{46}$$

  where $f : 2^V \rightarrow \mathbb{R}_+$, and concave_u is some non-negative non-decreasing concave function, and $m_u(a)$ is a modular function associated with feature u.

- The interaction between different features. The problem with this function is that the features themselves do not interact in any way, meaning that one feature u' might be partially redundant with another feature u''. Note a separate function could solve this, as in:

$$g(B) = \sum_{e \in E} \min(h_e(B), \alpha_e) \tag{47}$$

  where $g : 2^E \rightarrow \mathbb{R}_+$ is a submodular function that represents feature interaction, and h_e is a modular function defined on features, and E is a set of higher-level meta-features needing to be covered. Note that any function of the form $g(\alpha) = \min(\alpha, \alpha_u)$ is concave.

A solution to the issue addressed above is to use a construct we call *deep submodular functions*. Basically, we have a series of sets, $V = V^{(0)}$ which is the original (and true ‘‘ground’’) set (in the NLP community could be called the ‘‘surface’’ set), and also $V^{(1)}, V^{(2)}, \ldots, V^{(K)}$. The set V is the original ground set of items we wish to model (e.g., speech utterances, sentences, or bi-lingual sentence pairs). $V^{(1)}$ can be seen as a set of ‘‘features’’, $V^{(2)}$ as a set of meta-features (or ‘‘concepts’’, although that might not be the best choice of terms in a given applications community like NLP due to previous use of the term ‘‘concept’’), $V^{(3)}$ as a set of meta-meta features, and so on up a hierarchy of meta-features until $V^{(K)}$. Hence, we have a total ordering of these sets. Define $d^i = |V^{(i)}|$ as the size of set $V^{(i)}$.
Between successive pairs of sets, we have a matrix. Between layers $i-1$ and i let us call the matrix $w^{(i)}$, for $i \in \{1, 2, \ldots, K\}$ which is of dimension $d^i \times d^{i-1}$. Hence, rows of $w^{(i)}$ are indexed by

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

elements of $V^{(i)}$ and columns of $w^{(i)}$ are indexed by elements of $V^{(i-1)}$. Also, given $v^i \in V^{(i)}$, define $w^{(i)}_{v^i}$ to be the row of $w^{(i)}$ corresponding to element v^i, and $w^{(i)}_{v^i}(v^{i-1})$ is the element of the matrix $w^{(i)}$ at row v^i and column v^{i-1}. We will think of $w^{(i)}_{v^i} : V^{(i-1)} \to \mathbb{R}_+$ as a modular function defined on set $V^{(i-1)}$, thus this matrix gives us d^i such modular functions --- indeed, this is the reason for the notation $w^{(i)}_{v^i}(v^{(i-1)})$.
We can also think of this as a series of connected bipartite graphs where the right-hand side of one graph is the same as the left hand side of the next graph. The weights correspond to the edge weights of the bipartite graphs. Let g_{v^k} be some non-negative non-decreasing concave function (and the index v^k means we might have many of them, but we could just have only one of them). The class of deep submodular functions, in its most general form, for a K-layer model then becomes a function $f : 2^V \to \mathbb{R}_+$ defined as follows, for a given $A \subseteq V$

$$f(A) = \sum_{v^K \in V^{(K)}} g_{v^K}\left(\sum_{v^{K-1} \in V^{(K-1)}} w^{(K)}_{v^K}(v^{K-1}) g_{v^{K-1}}\left(\dots \sum_{v^2 \in V^{(2)}} w^{(3)}_{v^3}(v^2) g_{v^2}\Big(\sum_{v^1 \in V^{(1)}} w^{(2)}_{v^2}(v^1) g_{v^1}\big(\sum_{a \in A} w_{v^1}(a)\big)\Big)\right)\right) \tag{48}$$

As an example, for the function we wanted in the previous section, with $K = 2$, ground set V, feature set U, and meta-feature set E, using notation similar to what we used in previous sections, we would have:

$$f(A) = \sum_{e \in E} g_e\left(\sum_{u \in U} h_e(u) g_u\left(\sum_{a \in A} m_u(a)\right)\right) \tag{49}$$

Further instantiating, we can combine Equations 46 and 47 into one submodular function using:

$$f(A) = \sum_{e \in E} \min\left(\sum_{u \in U} h_e(u) \sqrt{\sum_{a \in A} m_u(a)}\,,\, \alpha_e\right) \tag{50}$$

3.2.3 Combinations of Polymatroid Functions.

Many operations preserve submodularity, some of which have been used implicitly in the above.

Conic or Convex Mixtures: Given a set of submodular functions $\{f_i\}_i$, we can form conic mixtures of the above. This means that

$$f(A) = \sum_i \alpha_i f_i(A) \tag{51}$$

is submodular as long as $\alpha_i \geq 0$.

Grouping Ground Set Elements: Similar to the way we grouped ground elements before, suppose we cluster the ground elements as follows $V = V_1 \cup V_2 \cup \dots \cup V_k$ into not-necessarily disjoint clusters V_i. We can define a function $f' : 2^{[k]} \to \mathbb{Z}_+$ as follows: For any $A \subseteq [k]$, define

$$f'(A) = f(\cup_{i \in A} V_i) \tag{52}$$

$f'(A)$ is submodular (and polymatroidal if f is).

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

''And''-like Functions: Suppose that we have two functions f_1 and f_2 and we want to find a set A such that both f_1 or f_2 are large, which is an ''and''-like construct. When the goal is to maximize, a sum of the two functions can be thought of as one form of and. Indeed, if $f = f_1 + f_2$ is large, and both $f_1, f_2 \geq 0$, then it is likely both f_1 and f_2 is large. Perhaps a more natural aggregation form would be $f = \min(f_1, f_2)$ which unfortunately is no longer submodular. However, there are bi-criterion approximations for aggregation functions of the form $f = \min(f_1, \alpha) + \min(f_2, \alpha)$ for various values of α that can be used and that preserve submodularity.

''Or''-like Functions Suppose that we have two polymatroid functions f_1 and f_2 and we want to find a set A such that either f_1 or f_2 are large, but we don't need them necessarily both to be large. Consider the following combination rule:

$$f(A) = f_1(A)f_2(V) + f_1(V)f_2(A) - f_1(A)f_2(A) \tag{53}$$

then f is also a polymatroid [14]. Moreover, if $f(A) > \tau$ then it must be that either $f_1(A)$ or $f_2(A)$ is large.

This section has presented a general overview of polymatroid functions. We next discuss which functions were chosen for the problem of speech and text data subset selection, and how they were instantiated.

Distribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

3.3 Instantiations of Submodular Functions for Data Selection

The submodular functions used in this work take the following general form:

$$f(A) = L(A) + \lambda R(A) \tag{54}$$

where the first term *L(A)* measures how well the selected feature *covers*, or *represents*, the original feature set; the second term *R(A)* measures how *diverse* the selected feature subset is; λ is a trade-off parameter. The notation is similar to a traditional machine learning optimization problem, where the first term measures the accuracy or fitness of the model, and the second term serves as a regularization term to control the complexity of the model to avoid overfitting. Two different classes of functions were utilized for *L(A)*: graph-based and feature-based functions. Regardless of which function is chosen, it is maximized by the same greedy selection algorithm shown in Figure 2.

Algorithm 1 Framework of GREEDY SELECTOR in pseudocode.

Given: Ground set $V = \{v_i\}$, a list of cost $C = \{c_i\}$ associated with the elements in V, $i = 1, \cdots, N$, a given budget B, and a similarity graph W where w_{ij} is the pairwise similarity between v_i and v_j.
Initialize $S \leftarrow \emptyset$, a priority queue $\rho \leftarrow \emptyset$
for $i = 1, 2, \cdots, N$ **do**
  $\delta \leftarrow f(v_i)$
  $\rho.push(tuple(i, \delta))$
end for
while $\sum_{i \in S} c_i \leq B$ **do**
  $k^* \leftarrow \rho.top().key$
  $\rho.pop()$
  $\delta = f(S \cup \{k^*\}) - f(S)$
  if $\delta > \rho.top().value$ **then**
    $S \leftarrow S \cup \{k^*\}$ {submodularity guarantees that $\delta \geq f(S \cup \{k\}) - f(S), k \in V \setminus S$ }
  else
    $\rho.push(tuple(k^*, \delta))$ {re-sort otherwise}
  end if
end while
return S

Figure 1: Greedy Selection Algorithm

3.3.1. Automatic Speech Recognition (ASR)

For ASR, both graph-based and feature-based submodular functions were explored. The main graph-based functions we tested were the facility location function (Equation 22, repeated here in slightly different form):

$$f_{fac}(A) = \sum_{i \in V} \max_{j \in A} w_{ij} \tag{55}$$

(where *S* is the subset, *V* is the ground set, and w_{ij} is the similarity between samples *i* and *j*) and the saturated graph-cut function (Equation 20):

$$f_{SC} = \sum_{i \in V} \min\{C_i(A), \alpha C_i(V)\} \tag{56}$$

(56)

with

$$C_i(A) = \sum_{j \in A} w_{ij}, \quad w_{ij} \geq 0 \tag{57}$$

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Four different similarity measures were explored in our work: Fisher kernel, string kernel, TF-IDF kernel, and rational kernel, all of which yield non-negative similarity scores.

***Fisher Kernel*:** The Fisher kernel [15] computes the similarity between speech utterances by taking into account the contribution of model parameters to the overall data likelihood. It first computes the vector of derivatives U_x of the log-likelihood of the acoustic data (X) with respect to the parameters in the phone HMMs ($1,2,...,m$ for m models):

$$U_X' = U_X^{\theta_1} \circ U_X^{\theta_2} \circ \ldots \circ U_X^{\theta_m} \quad \text{where } U_X^{\theta} = \frac{\partial \log P(X \mid \theta)}{\partial \theta} \tag{58}$$

Then the similarity score between two items i and j is computed as follows:

$$sim_{ij} = \max_{s,t} d_{st} - d_{ij} \tag{59}$$

with

$$d_{ij} = \| U_i' U_j' \|_1 \tag{60}$$

In initial experiments with the Fisher kernel it was observed that utterances judged as highly similar were usually not phonetically similar (i.e., they did not have the same or similar phonetic transcriptions) but resembled each other with respect to speaker, speaking rate, or other acoustic characteristics. In order to more specifically focus on phonetic similarity, three other similarity measures were investigated that belong to the class of discrete model-based similarity measures. These are based on a trained acoustic model (typically a hidden Markov model (HMM) or Gaussian mixture model (GMM)) that converts the sequence of acoustic feature vectors into a sequence of discrete symbols. These can be phonetic classes such as phones or phonemes, syllables, words, etc., or they can simply be indices of HMM states.

All of the similarity measures that are typically used for discrete sequences can then be applied to the resulting representation.

String Kernel: The string kernel [16] is a gapped, weighted subsequence kernel. Formally, a sentence s is defined as a concatenation of symbols from a finite alphabet (here: the inventory of phones) and an embedding function from strings to feature vectors, $\varphi:\Sigma^* \rightarrow H$. The string kernel function $K(s_i,s_j)$ computes the distance between the resulting vectors for two sentences s_i and s_j. The embedding function is defined as

$$\phi_{u(s)}^k = \sum_{i:u=s(i)} \lambda^{|i|} \quad u \in \Sigma^k \tag{61}$$

where k is the maximum length of subsequences, $|i|$ is the length of i, and λ is a penalty parameter for each gap encountered in the subsequence. K is defined as

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

$$K(s_i, s_j) = \sum_u \left| \phi_k^k(s_i), \phi_k^k(s_j) \right| w_u \tag{62}$$

where *w* is a weight dependent on the length of *u*. Finally, the kernel score is normalized by

$$\sqrt{K(s_i, s_i) * K(s_j, s_j)} \tag{63}$$

to discourage long sentences from being favored.

The more common (possibly gapped) subsequences are found in the two strings, the higher the similarity value. Figure 3 shows the algorithm for computing the string kernel.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Algorithm String Kernel

```
function K = DYNPROG(s,t,p,λ)
κ1 = zeros(length(s), length(t));
K(1) = 0; % length-1 co-occurrences
for i = 1 : length(s) do
   for j = 1 : length(t) do
      if s(i) = t(j) then
         κ1(i, j) = λ^2;
         K(1) = K(1) + κ1(i, j);
      end if
   end for
end for
K(1) = K(1)/λ^2; %renormalize
for l = 2 : p % co-occurrences of length 2, . . . , p do
   K(l) = 0;
   S(1 : (l − 1), 1 : length(t)) = 0;
   S(l : length(s), 1 : (l − 1)) = 0;
   for i = 1 : length(s) do
      for j = 1 : length(t) do
         S(i, j) = κ_{l−1}(i, j) + λS(i − 1, j) + λS(i, j − 1) − λ^2 S(i − 1, j − 1);
         if s(i) = t(j) then
            κ_l(i, j) = λ^2 S(i − 1, j − 1); K(l) = K(l) + κ_l(i, j);
         end if
      end for
   end for
   K(l) = K(l)/λ^{2l};% renormalize
end for
```

Figure 2: String Kernel Algorithm

***TF-IDF Kernel*:** The cosine similarity between sequences s_i and s_j is computed as

$$sim(s_i, s_j) = \frac{\sum_{w \in s_i, w \in s_j} tf_{w,s_i} tf_{w,s_j} idf_w^2}{\sqrt{\sum_{w \in s_i} tf_{w,s_i}^2 idf_w^2 \sum_{w \in s_j} tf_{w,s_j}^2 idf_w^2}} \quad (64)$$

where *tfw,si* is the count of n-gram *w* in *si* and *idfw* is the inverse document count of *w* (each sentence is a "document"). This similarity measure assigns greater importance to sequences occurring more rarely.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Algorithm TF-IDF Kernel

function K = TF-IDF-KERNEL(s,t, n)
$l_s = 0; l_t = 0; K = 0;$
for all n-gram w in s or t **do**
  Compute inverse document count of w as idf_w
end for
for all n-gram w in s **do**
  Compute term frequency count of w in s as $\mathrm{tf}_{w,s}$
  $l_s = l_s + \mathrm{tf}^2_{w,s}\,\mathrm{idf}^2_w$
end for
for all n-gram w in t **do**
  Compute term-frequency count of w in t as $\mathrm{tf}_{w,t}$
  $l_t = l_t + \mathrm{tf}^2_{w,t}\,\mathrm{idf}^2_w$
end for
$l_s = \sqrt{l_s}; l_t = \sqrt{l_t};$
for all n-gram w in both s and t **do**
  $K = K + \mathrm{tf}_{w,s}\,\mathrm{tf}_{w,t}\,\mathrm{idf}^2_w$
end for
$K = K/(l_s l_t)$

Figure 3: Algorithm for Computing TF-IDF Kernel

Rational Kernel: Instead of computing a pairwise similarity between two strings, it is possible to consider sets of strings or lattices by employing rational kernels [17]. Rational kernels are generalizations of the above kernels, and can be implemented efficiently using a finite-state transducer (FST) framework. For speech processing tasks, this means that rather than using a single sequence of phones or HMM states we can use lattices of symbols or N-best lists for each utterance, and compute the kernel between the lattices. The idea behind rational kernels is intuitive: two sequences, or more generally, two lattices, are similar when they share common subpaths. Assuming that lattices are represented by two weighted finite-state automata *A* and *B* a rational kernel can be defined as follows: a kernel *K* is *rational* if there exists a weighted finite-state transducer *T* such that for all weighted automata *A* and *B*, the following equation holds:

$$K(A,B) = \sum_{x,y} [[A]](x) \bullet [[T]](x,y) \bullet [[B]](y) \tag{65}$$

where *[[A]](x)* is the sum of the weights of all successful paths labeled with *x*, *[[T]](x,y)* is the sum of the weights of all successful paths with input *x* and output *y*, and • denotes the composition operation.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

In addition to graph-based functions with various kernels, feature-based functions were tested for ASR. These take the general form of

$$f(A) = \sum_{f \in F} g(\sum_{a \in A} m(f,a)) \tag{66}$$

where F is a feature set describing the utterance and A is the subset. The score $m(f,a)$ is a relevance score for feature f in utterance a and is typically instantiated to the TF-IDF weighted count of feature f in a in our experiments. The concave function g is instantiated to the square root function. The set of features is some inventory of phonetic units and can consist of words, phones, triphones, HMM states, Gaussian mixture ids, or any combination thereof.

Feature-based functions are less computationally complex since they do not require constructing a graph over the entire data set. In our experiments we have investigated both graph-based and feature-based functions for the small-scale task (TIMIT); for the large-scale Fisher task we have exclusively focused on feature-based functions.

Furthermore, feature-based functions can be defined over phonetic units trained either in a supervised or in an unsupervised manner. When supervised information (i.e., audio transcriptions) is available, the phonetic units may consist of the (true) words, syllables, context-dependent phones, or context-independent phones derived from the transcriptions. It would be preferable, however, to not rely on supervised transcriptions – in that case, our approach could be applied even to new languages or tasks for which no transcriptions are available. The phonetic tokenization could then be obtained either by applying a phone tokenizer trained on a different language/task, or by unsupervised training of acoustic models on the data of interest. Our initial experiments utilized models trained from supervised transcriptions, which represents the best-case scenario. For the small-scale ASR task (TIMIT) we have additionally investigated unsupervised acoustic modeling, which has yielded comparable results (see Section 4.1).

3.3.2. Machine Translation

The large machine translation data sets considered in this project contain several million sentences pairs. Therefore, graph-based functions were not considered for MT data selection since they require constructing a graph of size $|V|^2$ for $|V|$ sentences. Instead, feature-based functions of the exact same type as those used for ASR are considered. The basic form is identical, only the definitions of the feature set F and the relevance score $m(f,a)$ change. The feature set F consists of word sequences of up to length n (n-grams or phrases) that are found in the development and test data. For the relevance score a large number of options were investigated, including raw frequency counts, TF-IDF weighted frequency counts, weighting by n-gram length, weighting by the ratio of counts in the test vs. training set, etc. The best final function was determined to be

$$f(A) = \sum_{f \in F} \sqrt{\sum_{a \in A} tfidf(f,a) * \frac{c_{test}(f)}{c_{train}(f)} * \alpha^{|f|}} \tag{67}$$

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Here, *tfidf(f,a)* is the TF-IDF weighted count of feature *f* in *a*, *c(f)* is the raw count, and α is a length weight. Thus, the TF-IDF weighted count is further weighted by the ratio of raw counts in the test vs. training data as well as by a feature length specific weight.

The various weighting factors in this function were inspired by previous work and/or insights into the properties of machine translation systems -- e.g., tf-idf weighting worked well in previous work on submodular text summarization (). Moreover, it is widely known in the MT community that longer n-grams typically lead to more fluent translations and are weighted more highly during the computation of the BLEU evaluation criterion. The ratio of test counts to training counts for a particular n-gram mimics the use of in-domain (test set) language models and out-of-domain (training set) language models in the modular cross-entropy data selection method by (Moore & Lewis, 2010). Further aspects of the function, such as the use of a square root function rather than a different concave function, and the particular value for α, were determined empirically. Thus, establishing a good submodular function is a combination of system-specific insights and empirical experimentation. In any application the key to finding a good function is to find to a good surrogate function for the eventual evaluation criterion.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

4.0 RESULTS AND DISCUSSIONS

4.1 ASR Experiments

The submodular data selection framework was evaluated on two tasks. Our main task is large-vocabulary speech recognition on conversational English telephone data (called "Fisher task" for short). For this purpose, we use the Decipher speech recognition system provided by our project partner, SRI International. Since the training time for this task/system is long (five to seven days when using the complete data set), a simpler task, viz. phone recognition on the TIMIT corpus, was selected in order to more rapidly explore and compare different submodular functions. We first provide a description of the corpora and recognition systems and then report on the most relevant and insightful experiments for each task.

4.1.1. Data and System Descriptions

TIMIT Task: The TIMIT task consists of phone recognition on the TIMIT database [18], a phonetically balanced speech corpus in studio-quality speech. This task focuses on acoustic modeling only, thereby excluding potential language modeling effects. The sizes of the training, development and test data are 4620, 200 and 1144 utterances, respectively. The test set is the complete TIMIT test set minus the *sa* sentences, and the 200 utterances used for the development set.

Preprocessing was done by extracting 39-dimensional MFCC feature vectors every 10 ms, with a window of 25.6ms. Speaker-dependent mean and variance normalization was applied to the feature vectors. Phone recognition experiments were conducted both with supervised models (trained from the manual phone transcription provided with the corpus) and unsupervised models (trained to maximize the likelihood of the data, without any transcriptions). For the first set of experiments, a supervised 3-state monophone HMM system is trained (48 phone class). The HMM state output distributions are modeled by diagonal-covariance Gaussian mixtures with the number of Gaussians ranging between 4 and 64, depending on the data set size. This is a simple acoustic model that was chosen over a more computationally complex model in order to ensure quick experimental tuning times. Following the selection of subsets (1%, 2.5%, 5%, 10%, 20%, 30% and 40% of the data, measured as percentage of non-silence speech frames) the phone recognizer was trained on the resulting sets; the performance was evaluated on the test set of 1144 utterances, collapsing the 48 classes into 39 in line with standard practice in the community. Performance is reported as phone recognition accuracy. The phone recognition accuracy obtained by the baseline system using 100% of the training data is 63.28%.

Fisher Task: For large-scale ASR experiments the SRI Decipher system was used, which was provided to UW by SRI International as the subcontractor on this project. The ASR system was trained on 1300 hours of acoustic data taken from the Switchboard, Switchboard Cellular, and Fisher corpora.

The test sets are the NIST Rich Transcription development sets from 2001 and 2002, with 2.2 hours and 6.3 hours of acoustic data, respectively. All audio files had a 8 kHz sampling rate. The Decipher recognizer uses 13 mel-frequency cepstral coefficients (MFCC) as signal processing features, along with their 1^{st}, 2^{nd}, and 3^{rd} order derivatives. The resulting 52-dimensional feature vectors were mean and variance normalized, and heteroscedastic Linear Discriminant Analysis (HLDA,[19]) was used to reduce it to a 39-dimensional feature vector. The features were then

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

discriminatively transformed using feature minimum phone error training (fMPE) [20]. The recognizer used a three-state left-to-right HMMs with GMMs as output probability distributions. Each GMM represented a decision-tree clustered cross-word triphone. Initial GMMs were estimated with maximum likelihood optimization criterion, and these initial models were used to generate phone lattices. The lattices were used in minimum phone error training (MPE) to create the final GMMs [21]. During decoding, a first pass search is performed using a bigram language model. A recognition pass using maximum-likelihood linear regression (MLLR) speaker-adapted acoustic models generates a set of lattices. Finally, these lattices were rescored with a trigram language model to generate the final output. The language models (LMs) were estimated by interpolating various genre-specific LMs; the genres were transcriptions of meetings, spontaneous telephone speech, and broadcast news. Additionally, LMs were trained using web data selected to be similar to be similar to the transcription data. The interpolation weights were optimized on a held out set of meeting data. The entire system's training and decoding architecture is shown in Figure 5. The acoustic front end, training, and recognition were done using SRI's DECIPHER system (Stolcke et al., 2000). The language model training and lattice rescoring were accomplished using the SRI language modeling toolkit.

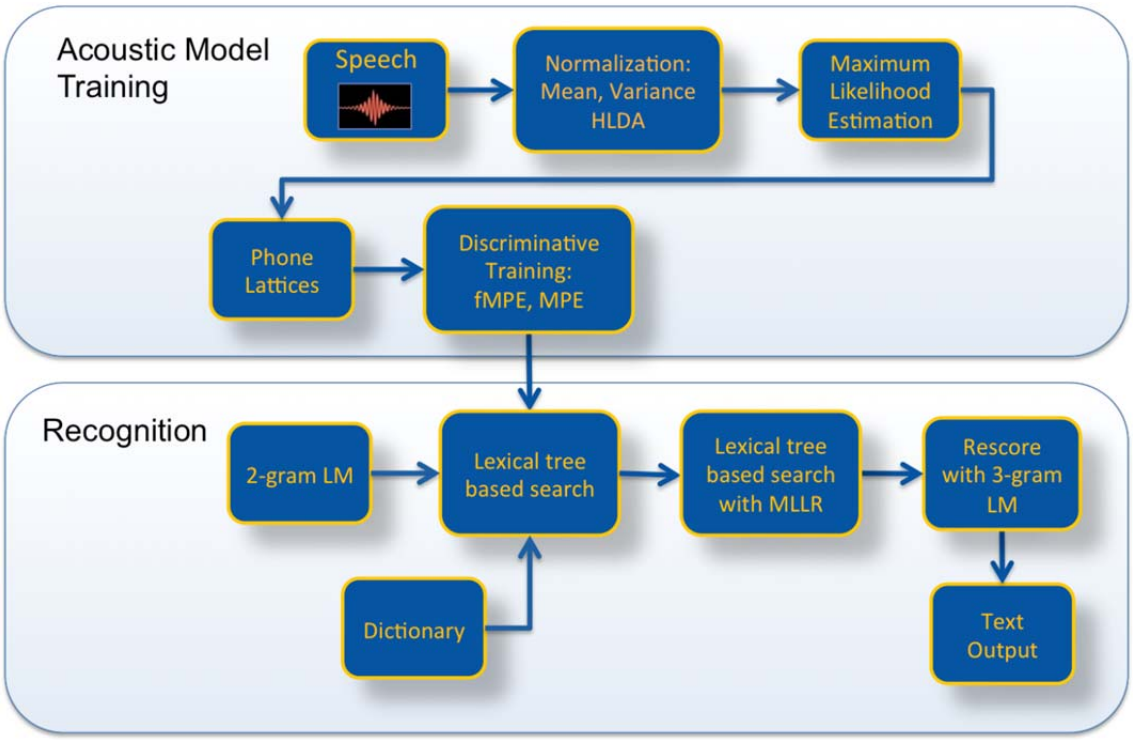

Figure 4: SRI Decipher ASR System - Training and Decoding Scheme

4.1.2. Experiments on the TIMIT Task

For the baseline experiments that utilize random data subset selection 100 different subsets of the desired size were drawn randomly from the entire training set. The phone recognition accuracies were averaged over all 100 draws. The second baseline method is the method described in [4], which aims to maximize the entropy of the distribution over phones in the selected subset. The procedure has the following steps::

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Algorithm Histogram-entropy Greedy Search

```
for all utterance u_i do
    if Adding u_i increases entropy of distribution over phones by some threshold e then
        add u_i
    end if
end for
```

Figure 5: Histogram-Entropy-Based Data Selection Algorithm

Next the first three similarity measures (Fisher kernel, TF-IDF kernel, and string kernel) were tested in combination with the two graph-based submodular functions. The parameters of the kernels (i.e. the kernel order, gap penalty, and contiguous substring length of the string kernel) were optimized on the development set. The best values were 0.1 for gap penalty, 4 for kernel order, and 3 for substring length. Figure 7 shows the performance of the facility location function with the different kernels, as well as the random and histogram-entropy baselines.

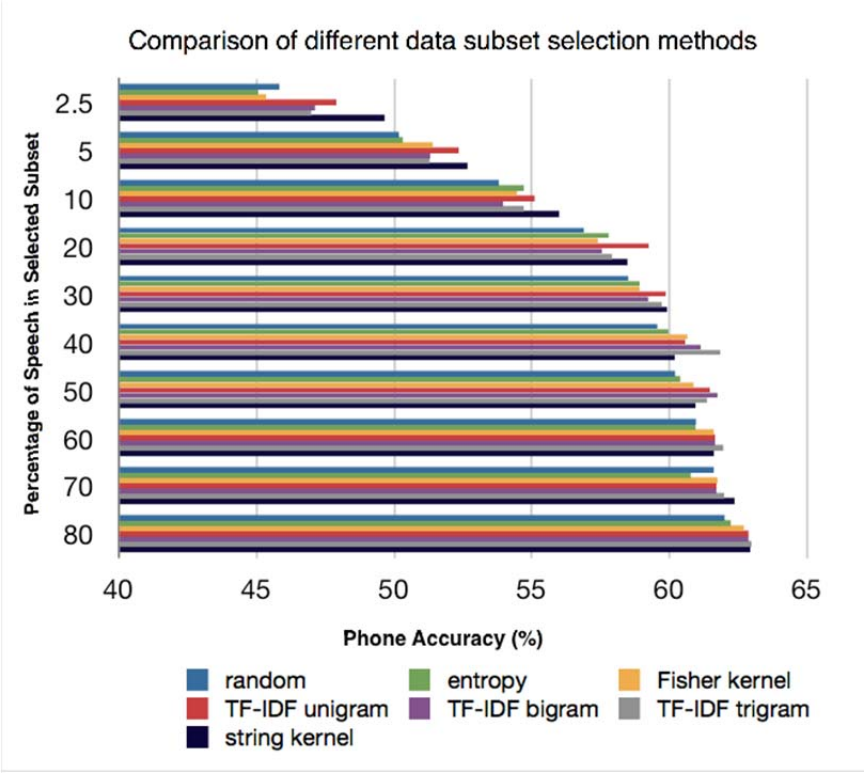

Figure 6: TIMIT Phone Accuracy for Different Subset Sizes and Similarity Measures

(Facility Location Function)

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

In these experiments the histogram-entropy baseline outperformed random selection; the submodular functions outperformed both significantly ($p < 0.05$). The Fisher kernel was generally worse than the TF-IDF or string kernel. The best performance was obtained with the string kernel, especially on small data sets (2.5% - 10%). Experiments with the fourth similarity measure, the rational kernel applied to n-best decoded phone sequences (*n=10*), did not yield any further improvements over the string kernel. This may have been due to the fact that the n-best decoded phone sequences were very similar and did not offer enough variation to take advantage of the rational kernel. A comparison of the facility location function and the saturated graph-cut function (Figure 8), both using the string kernel, showed that the facility location function yields superior results on this task.

Since the histogram-entropy method uses the true transcriptions (as in [4]), another experiment was conducted to determine the difference between using the true transcriptions vs. decoded phone sequences for submodular data selection. In the latter case, phone models are first trained on the transcriptions and then used in an unconstrained decoding pass to generate phone sequences. Figure 9 shows the results. The hypothesized phone sequences in fact lead to better results. This may be because they are a product of both the acoustic signal and the underlying phonetic sequence, whereas the true transcriptions only provide information about the phonetic sequence but ignore the acoustic properties of the signal.

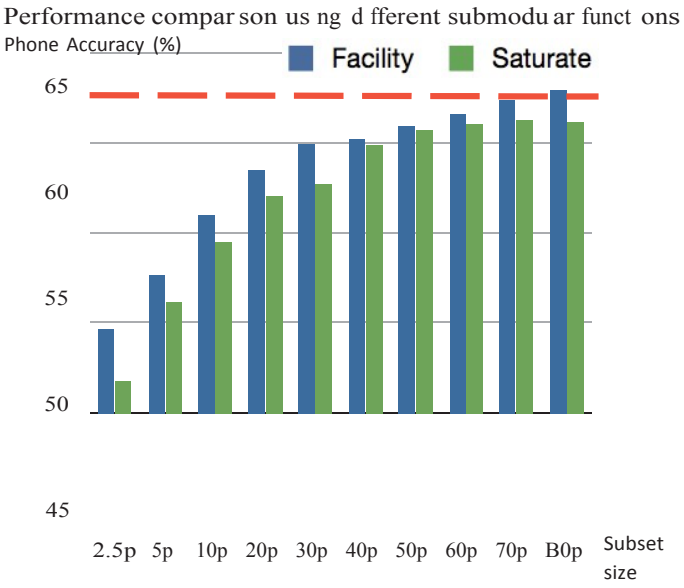

Figure 7: Comparison of TIMIT Phone Accuracies
(Facility Location Function vs. Saturated Graph Cut Function.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

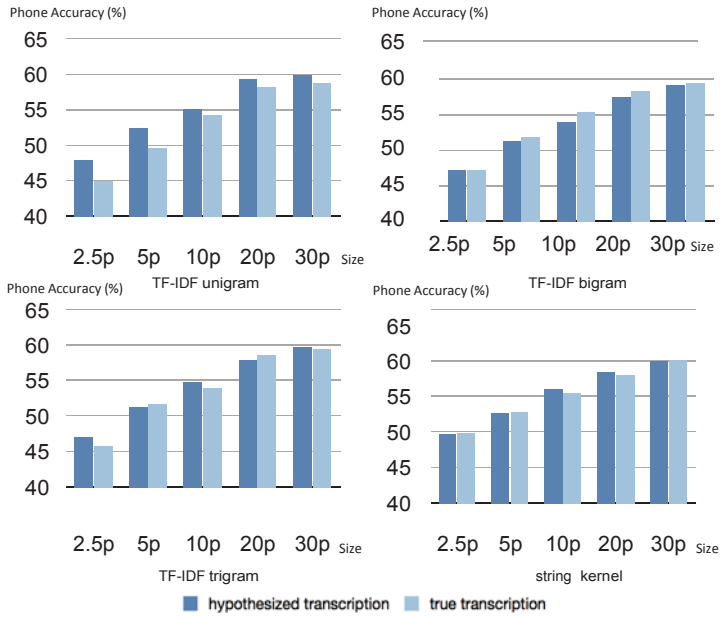

Figure 8: TIMIT Phone Accuracies Obtained with True vs. Hypothesized Phone Sequences

Next, a diversity function *R(S)* was added to the coverage function to further reduce redundancy in the selected subset. This function takes the following form:

$$R(S) = \sum_{n=1}^{N} \sqrt{\sum_{j \in P_n \cap S} \sum_{i \in V} \frac{w_{ij}}{|V|}} \tag{68}$$

where g is a concave function and $P_1,\ldots,P_N$ is a partitioning of the ground set into N clusters, obtained by k-means clustering. This function encourages selecting items from different clusters. Table 1 shows a comparison of results with and without diversity term. The diversity term is helpful especially at small data set sizes.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 1: Comparison of Phone Accuracy Rates on TIMIT

(Facility Location Function with and without Diversity Term)

Data set size (%)	Random	Histogram Entropy	Facility Location	Facility Location+Diversity
1	37.98	35.99	38.80	42.73
2.5	45.93	46.47	48.46	49.44
5	50.58	51.04	52.62	52.82
10	54.37	55.72	56.10	56.32

In addition we investigated convex mixtures of different functions. In this set of experiments, we choose two submodular objective functions for combination: the facility location function (f_1) and a feature based submodular function (f_2). We then take the convex combination of $f_1$and f_2 as

$$f = \theta f_1 + (1-\theta) f_2 \tag{69}$$

f_1 uses the facility location function with string kernel similarity on supervised decoded phone sequences, and the features in f_2 are the TF-IDF weighted triphones from the true phone transcriptions. Note that both f_1 and f_2 are normalized by function values at ground set respectively. All combinations of discretized parameters are tuned on the development set for each percentage case and then evaluated on the full test set excluding the development set. Notice that the performance numbers in the following table are not comparable to all our previous results, since for different subset sizes, the model complexity was fixed in this experiment (all as 8-component GMM-HMM). In the previous experiments the model complexity was increased for increasing amounts of data. In the following table the performance of the convex combination is compared against the performance of either component function in isolation. The performance of the tuned convex combination of submodular functions outperforms both the isolated functions at some of the small percentage cases, except at 2.5%.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 2: Phone Accuracy Rates on TIMIT for Component Functions and Convex Combination

Data Set Size (%)	Random	Tuned Convex Combination Of F_1 And F_2	Facility Location Function F_1	Feature Based Submodular Function
1	24.83	35.03	34.27	30.88
2.5	41.26	45.18	46.35	42.70
5	49.98	51.34	51.03	50.95
10	54.42	55.43	55.42	54.87
20	56.23	56.54	56.55	56.19
30	57.11	57.32	57.04	57.12
40	58.01	58.38	58.38	57.80

The final set of TIMIT experiments reported here utilizes entirely unsupervised phone models and 2-layer "deep" submodular functions. That is, the true transcriptions were not used at all in training. Instead an unsupervised GMM was trained to maximize the data likelihood.

The two-layer feature-based function is defined as follows:

$$f(A) = \sum_{u \in U} \sqrt{\sum_{f \in F} w_{uf} \sum_{a \in A} m(f,a)} \tag{70}$$

Here, there are two feature sets: a set of high-level features U and a set of low-level features F.

This function is used in combination with k-component GMMs: a 512-component GMM is used to generate low-level features, and a 32-component GMM is used to generate high-level features. In both cases, the Gaussian mixture indices are used as the features, either in isolation (unigrams), or considering sequences of two indices (bigrams). To further speed up the training process, *k-means++* is applied to the original acoustic features; the initial parameters of both GMMs are the centroids from the outputs of *k-means++* algorithm and both GMMs are trained with 20 EM iterations. The weights w are the co-occurrences of features of sets F and U on the training set.

This selection function was evaluated on the TIMIT dataset. In Figure 10, the first two bars show the performance of random selection and the histogram entropy method; the 3rd bar is the supervised trained HMM where the features consist of phone sequences obtained using forced-alignment of the training transcriptions. The 4th and 5th bar show the performance of an unsupervised GMM system with 1-layer and 2-layer feature based functions (unigram features); the 6th and 7th bar show the performance of and unsupervised GMM system with 1-layer and 2-

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

layer feature based functions (bigram features). The performance of the best unsupervised systems is either equivalent to that of the best supervised system or even outperforms it.

This shows that it is in principle possible to achieve superior results even in the absence of any transcription or annotation of the acoustic data.

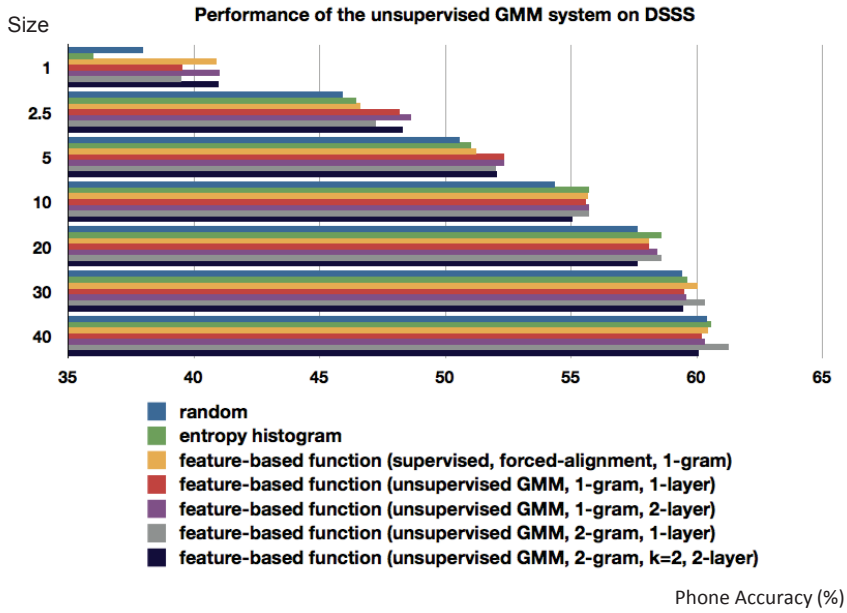

Figure 9: TIMIT Phone Accuracies: Supervised vs. Unsupervised Acoustic Models

4.1.3. Experiments on the Fisher Task

For the Fisher task we constructed a random baseline by randomly sampling four data sets of the desired sizes for each condition (1%, 5%, 10%, 20%). The histogram-entropy baseline systems were constructed in two different ways: (a) using the words from the true word transcriptions as phonetic units, and (b) using the triphone state labels from a forced alignment of the transcriptions to the acoustic data. The first method is the one used in [4]. One issue that was encountered on the Fisher data is that the objective criterion of this method (maximum entropy of the distribution over phonetic units in the data) may saturate (i.e. increase no further) before the budget constraint is reached. This was the case for the 20% condition – in this case, the entropy saturates before 20% of the data have been selected. To reach 20%, further data would have to be added randomly, which would render the method largely equivalent to the random selection baseline. Therefore no results are available for this method for the 20% case. Table 3 shows the word error rates for the random and histogram-entropy baseline systems.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 3: Word Error Rates on Fisher Task (Baseline Systems)

Data Subset Size	1%	5%	10%	20%	100%
Random	52.1	38.2	35.1	34.4	31.0
Entropy (words)	49.6	36.5	34.8	N/A	
Entropy (triphones)	47.5	37.6	34.2	N/A	

Initially, several variants of the graph-based submodular objective functions were explored for the Fisher task, using the TF-IDF kernel as the similarity measure, which has lower computational complexity than the string kernel. In the first set of experiments, the pair-wise similarity graph was constructed using a TF-IDF bigram kernel on the triphone labels resulting from the forced alignments. The ground set size is about 1.3 million utterances, hence it is very expensive to store the dense affinity matrix on disk, which requires $O(n^2)$ memory for storage. Instead, we store the affinity matrix as a k-nearest neighbor graph (k-NNG), where $k=1000$. The k-NNG is defined as the spanning sub-graph where each vertex is connected only to its k most similar neighbors. The graph-based submodular function is chosen as the facility location function, and the exponent hyperparameter for normalizing by cost during greedy selection is set to be 0.2. Initial results of this method fell in between those of the two histogram-entropy baselines.

It was decided that graph constructions presented a significant computational bottleneck for further experimentation; all successive experiments were therefore carried out with the feature-based function. The different variants of the feature-based function involved different phonetic units (words vs. triphones vs. triphone HMM states), n-grams of phonetic labels with various instantiations of n, different ways of normalizing feature counts (e.g., TF-IDF), automatic clustering and relabeling of phonetic labels according to short, medium, and long duration, and labels derived from unsupervised acoustic models. In addition it was discovered that it is important to optimize the complexity of the acoustic model (number if initial clusters for bootstrapping the acoustic model, and the number of leaves in the decision tree used for state clustering) for each data set selected. This is important because data sets with greater inherent variability can support more parameters in the acoustic model, whereas homogeneous data sets used with a large number of parameters will lead to poorly trained parameters. The number of parameters was optimized on the development set. The best results obtained with submodular selection were based on a feature-based function with TF-IDF weighted counts of triphones resulting from forced alignment of the word transcriptions, with a knapsack scaling factor of 0.2. Results obtained with this method outperform almost all baseline systems (Table 4).

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 4: Word Error Rates on Fisher Task (Baselines and Best Submodular Method)

Size	1%	5%	10%	20%	100%
Random	52.1	38.2	35.1	34.4	31.0
Entropy (words)	49.6	36.5	34.8	N/A	
Entropy (triphones)	47.5	37.6	34.2	N/A	
Submodular (triphones)	**47.5**	**35.7**	**33.3**	**32.6**	

In another set of experiments the Decipher system was replaced by a more advanced system based on deep neural networks for acoustic modeling that is being developed at SRI. Table 5 shows the results. The random baseline represents the average over three different random systems. For all systems, the number of parameters (layers in the deep neural network) was optimized, and the best result obtained on the development set was selected. Random selection occasionally outperforms the histogram-entropy method (due to having only a single random set). The submodular method outperforms both baseline methods.

Table 5: Word Error Rates (%) Obtained with Deep Neural Network System and Different Selection Methods

Size	Random	Hist-Entropy	Submodular
1%	43.7	42.8	**41.1**
5%	34.3	33.9	**31.8**
10%	31.5	31.3	**29.3**
20%	29.8	N/A	**28.2**
100%	26.0		

Finally, acoustic models trained in an unsupervised way were investigated. HMMs with 40 states and a total number of mixture components ranging from 64 to 4096 were trained using k-means, and the resulting trigrams over HMM states were used as features. However, results were negative in that they did not exceed the random baseline results. It is likely that the phonetic resolution in this system was not high enough – for comparison, the supervised GMM system has an order of magnitude more mixture components (about 100k). A parallel k-means implementation has been developed to be able to train an unsupervised system with 100k components. Future experiments will evaluate this system on the Fisher task.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

4.2 Machine Translation Experiments

For machine translation, the "selection-for-adaptation" scenario described in the Introduction was adopted, due to the fact that this is a problem of great interest to current machine translation research and several approaches to this problem have been reported in the literature. The goal is to find a training data subset that matches a given test set as well as possible, thus producing good results on that particular set. A common approach to this problem is the cross-entropy based data selection method [1,2]. As explained above, this is a modular method that seeks to select sentences that receive a high score by an in-domain language model but a low score by a generic out-of-domain language model.

For our purposes the cross-entropy baseline was implemented as described in [1], using the following steps:

1. A language model is trained on the "in-domain" data set – this is the combined development and test data.
2. The "out-of-domain" data set (i.e. the training set) is shuffled randomly, and a subset of approximately the same size as the in-domain data set is selected. Then a language model is trained on this set, using only the vocabulary obtained from the in-domain data set.
3. Both language models are applied to the entire training data, scoring each sentence with both models and producing two log-probability scores ppl_{id} and ppl_{ood}.
4. For each sentence $v \in V$ a relevance score is computed as $S(v) = ppl_{id} - ppl_{ood}$, and all sentences are ranked according to their relevance scores.
5. Finally, the top N sentences are selected such that their combined word count does not exceed the budget (desired data set size).

In addition to this method, random data subset selection is used as another baseline. For this method, several random subsets are selected and the resulting system performance numbers are averaged over all trials. System performance is evaluated using two evaluation criteria: BLEU and position-independent word error rate (PER).

4.2.1. Data and System Descriptions

***NIST Task*:** Our primary machine translation task (the "NIST" task) utilizes approximately 189M words of parallel Modern Standard Arabic (MSA) and English text data. The list of corpora is provided in Table 1; all corpora are available through the Linguistic Data Consortium (LDC) and were admitted as training material in the most recent (2012) NIST Machine Translation benchmark evaluations. The corpora were preprocessed by extracting text from the xml/html documents, removing meta- characters, non-sentence punctuation, non-UTF-8 characters, etc., and converting numbers, dates, and URLs into variables. Preprocessing on the source side was aided by the MADA tool [23] for conversion to Buckwalter format and tokenization; additional Arabic preprocessing and all English preprocessing was done by in-house tools.

The development data consisted of the NIST MT 2006 test set; the evaluation data consisted of the NIST MT 2009 test set and the NIST MT 2012 progress test set. All of these sets include two different text genres, newswire data (nw) and web data (wb). The sizes of the sets are shown in Table 6.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 6: Sizes of Data Sets Used for the NIST MT Task

Set	Number of Sentences	Number of English/MSA Words
Training	5,026,387	189M / 145M
MT06 (Development)	1,529	48k / 44k
MT09 (Test 1)	1,313	49k / 44k
MT12 (Test 2)	1,378	56k / 51k

The language model training data consists of the English side of the parallel training data and, additionally, the English text corpora listed in Table 7. These were preprocessed in the same manner as the English sides of the parallel corpora. The language modeling corpora were then divided into three clusters: GALE data, Gigaword, and all remaining corpora. A separate 5-gram language model was trained on each cluster using the SRILM tool. The smoothing technique was Witten-Bell discounting in combination with interpolation of higher-order and lower-order n-grams. The language model vocabulary was limited to the set of words occurring on the English side of the machine translation training data. The three separate language models were then interpolated; interpolation weights were optimized separately on the newswire and the web data portions of the development data, thus resulting in two different language models. These language models remained fixed for all experiments, i.e. they were not retrained, in order to better assess the impact of different training data sets on the performance of the system. During decoding, the appropriate genre language model was used for the corresponding portion of the development/test data.

The translation system was trained using the GIZA++/Moses infrastructure [24]. Word alignments were trained with the mgiza, the multi-core version of GIZA++. The translation model is a non-hierarchical phrase-based log-linear model with a maximum phrase length of 7. More complex models, such as hierarchical phrase-based or syntax-based models were excluded deliberately in order to limit the experimental turn-around time needed for each experiment. The reordering model is a hierarchical model according to [25] (Moses option *–hier-msd-bidirectional-fe*). The feature weights in the log-linear model were optimized to maximize the BLEU score on the development set using minimum error rate training (MERT) as implemented in Moses.

Table 7: List of Language Modeling Corpora in the Aribic-to-English NIST Task

LDC Calalog Number	Size (Million Words)	Name
LDC2011T07	1,756	English Gigaword
LDC2002T31	477	AQUAINT
LDC93T1	31	ACL/DCI
LDC2005T35	22	American National Corpus (ANC)
LDC2005T29	257	HARD

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Postprocessing involved recasing the translation output, replacing variable names with their original corresponding tokens, and normalizing spelling and stray punctuation marks. The recasing model is another MT system without reordering, trained on parallel cased and lowercased versions of the training data. Similar to the language model the recasing model remains fixed for all experiments and is not retrained for different sizes of the training data.

Evaluation was done by computing the BLEU and NIST scores using the official NIST evaluation tool *mteval-v13a.pl* with the –c flag for case-sensitive scoring.

Transtac Task: Similar to the ASR experimental setup a smaller data set was identified for preliminary experiments in order to rapidly test different submodular functions and debug experimental procedures. This data set consists of parallel Iraqi Arabic-English Transtac data that is currently used in the DARPA BOLT project to develop a spoken dialog system for this language pair.

The data was provided to us by our project partner SRI International. The sizes of the training, development and evaluation sets are shown in Table 8.

Table 8: Data Set Sizes for the Transtac MT Task

Set	Number of Sentences	Number of Arabic/English Words
Training	766,080	6,946,684 / 7,284434
Development	3,430	40,511 / 43,282
Evaluation	3,097	39,711 / 41,145

For the development and evaluation sets one target reference translation was provided. The data used for language modeling consists of the English side of the training data. The data was preprocessed using tools developed in the DARPA BOLT project, which includes punctuation removal, tokenization, and (for Arabic) conversion to Buckwalter format and morphological segmentation.

The translation system trained on this data used the same settings as the system trained on the NIST task (see above). The language model is a 5-gram model with Witten-Bell discounting and interpolation of higher and lower-order n-grams. Postprocessing involves uppercasing, de token-ization and spelling normalization.

4.2.2. Experiments on the Transtac Task

Two baseline systems were trained on the full training set, one for the English-to-Arabic translation direction, and another for the reverse direction. The baseline BLEU/PER scores are 32.5/42.3 for the Arabic-to-English translation direction and 16.2/58.5 for the reverse direction. For the random baseline experiments, four data subsets of the desired size were selected randomly from the full training data set, and the BLEU/PER scores of the resulting systems were averaged. For the cross-entropy baseline, language models of order 3 were trained as described above.

Table 9 shows the BLEU/PER scores of the two baseline methods. The cross-entropy based method outperforms random selection and approximates the performance (as measured by BLEU) of the baseline system trained on the full data set at about 40%.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 9: Baseline BLEU (%)/PER Scores on Transtac Task (Arabic-to-English)

Data Set Size	Random	Cross-Entropy
10%	28.1 / 46.7	30.2 / 44.3
20%	29.4 / 45.5	31.5 / 43.2
30%	30.4/ 44.4	31.8 / 42.7
40%	30.7 / 44.6	32.2 / 42.4
100%	32.5 / 43.2	

Baseline results for the English-to-Arabic translation direction are shown in Table 10. Again, the cross-entropy selection method outperforms random selection; moreover, performance matches the baseline system at 10% and even exceeds it at 20% or higher. This indicates that much of the training data is either redundant or not relevant to the test set and that a much more focused and accurate system can be trained when data subset selection is applied.

Table 10: Baseline BLEU (%)/PER Scores on Transtac Task (English-to-Arabic)

Data Set Size	Random	Cross-Entropy
10%	15.0 / 61.1	16.1 / 58.5
20%	16.1 / 59.7	17.1 / 57.7
30%	16.2 / 59.5	17.4 / 57.5
40%	16.0 / 59.6	17.3 / 57.8
100%	32.5 / 43.2	

Next, a wide range of submodular selection methods was implemented and evaluated for the Arabic-to-English translation system. Different instantiations were distinguished by their values for parameters in Equation (67), in particular whether or not TF-IDF weighting was used, whether the ratio of test vs. training counts was used, the value of the phrase length dependent weight, and the weight for the knapsack constraint. Table 11 shows the results for the Arabic-to-English translation direction.

Table 11: Comparison of BLEU(%)/PER Scores on Transtac Task (Arabic-to-English)

Data Set Size	Random	Cross-Entropy	Best Submodular System
10%	28.1 / 46.7	30.2 / 44.3	**32.3 / 43.2**
20%	29.4 / 45.5	31.5 / 43.2	**32.5 / 42.5**
30%	30.4 / 44.4	31.8 / 42.7	**32.5 / 42.5**
40%	30.7 / 44.6	32.2 / 42.4	**32.6 / 42.1**
100%	32.5 / 43.2		

The best submodular data selection method found for the Arabic-to-English translation direction was then also applied to the English-to-Arabic system. Results are shown in Table 12. The submodular system outperforms the baseline systems by a substantial amount, even in the lowest percentage cases.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 12: Comparison of BLEU(%)/PER Scores on Transtac Task (English-to-Arabic)

Data Set Size	Random	Cross-Entropy	Best Submodular System
10%	15.0 / 61.1	16.1 / 58.5	**17.9 / 57.5**
20%	16.1 / 59.7	17.0 / 57.7	**17.7 / 57.3**
30%	16.2 / 59.5	17.4 / 57.5	**17.5 / 57.4**
40%	16.0 / 59.6	17.3 / 57.8	**17.5 / 57.3**
50%	16.5 / 58.6	17.0 / 57.9	**17.5 / 57.3**
100%	16.2 / 58.5		

4.2.3. Experiments on the NIST Task

For the NIST task, we constructed random baselines by randomly sampling three data sets for each of the desired sizes (10%, 20%, 30%, and 40% of the full training data set).

MT systems were trained on each set, and BLEU scores were averaged over all three samples.

Table 13: BLEU Scores on MT09 Set

Data Set Size	Random (stddev)	Cross-Entropy	Submodular
10%	0.3818 (+/- 0.001)	0.4043	**0.4140**
20%	0.3931 (+/- 0.003)	0.4111	**0.4129**
30%	0.3947 (+/- 0.002)	0.4155	**0.4179**
40%	0.4101 (+/- 0.002)	0.4186	**0.4187**
100%	0.4113		

Table 14: BLEU Scores on MT12 Set

Data Set Size	Random (stddev)	Cross-Entropy	Submodular
10%	0.3460 (+/- 0.001)	0.3758	**0.3798**
20%	0.3592 (+/- 0.002)	0.3748	**0.3785**
30%	0.3621 (+/- 0.003)	0.3788	0.3764
40%	0.3750 (+/- 0.003)	0.3836	0.3810
100%	0.3818		

The cross-entropy baseline selection method was applied as described above in the Transtac task. The cross-entropy method outperforms the random baseline by 2-3 BLEU points at low percentage cases; the gap narrows as more data is selected. Results show again that a subset of 30-40% is sufficient to replicate or even surpass the performance obtained with 100% of the training data. The best submodular data selection method outperforms both baselines under all conditions on the MT09 set; the difference is larger at smaller data subset sizes. The performance of the baseline system at 100% is outperformed by all submodular subsets, even when using only 10% of the data. This indicates that much of the training data is redundant or irrelevant to the test set; a reduction of at least 90% can be achieved (subsets smaller than 10% were not evaluated). On the MT12 set improvements over the baseline methods are obtained in the 10% and 20% cases but not at the higher percentages. The reason for this might be that the system was not sufficiently optimized under these conditions. An analysis of the phrase table showed that the submodular system did in fact achieve better coverage of the test set (on both the source and the target side) than the cross-entropy system, as expected. However, this did not translate into

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

improvements in BLEU. A possible problem may have been the weight optimization (minimum error rate training) procedure – the number of iterations in this module was capped at 10 to reduced experimental run time. This may not have been sufficient to optimize the larger models obtained in the 30% and 40% percentage cases – note that these are the largest/most diverse models overall in our MT experiments since the MT12 set is larger than the MT09 set, thus resulting in larger phrase tables. Additional experiments will be conducted in the near future to examine this issue.

Further experiments were conducted to determine how the cross-entropy and submodular methods are affected by different n-gram orders/different maximum phrase length. To this end cross-entropy based data subset selection was carried out multiple times, each time with a different-order language model (n = 3,5, and 10). The submodular selection method was run with different maximum phrase lengths (maximum |f| = 5, 7, and 10). Tables 15 and 16 show the results.

Table 15: Results of Cross-Entropy Method with Different N-gram Orders

Data Set Size	n=3	n = 5	n = 10
	MT09 / MT12	MT09 / MT12	MT09 / MT12
10%	**0.4079 / 0.3758**	0.3941 / 0.3593	0.3890 / 0.3595
20%	**0.4111 / 0.3748**	0.3956 / 0.3633	0.3969 / 0.3620
30%	**0.4155 / 0.3788**	0.3983 / 0.3652	0.4016 / 0.3660
40%	**0.4186 / 0.3836**	0.3943 / 0.3705	0.3943 / 0.3725

The cross-entropy method does not benefit from higher n-gram orders. This is arguably due to the fact that the LMs in this method are trained on fairly small data sets (the dev/test sets, and an equivalently-sized portion of the training set). LMs with larger orders may therefore be undertrained and rely on back-off to lower-order n-grams.

Table 16: Results of Submodular Method with Different Maximum Phrase Lengths

Data Set Size	|f| = 5	|f| = 7	|f| = 10
	MT09 / MT12	MT09 / MT12	MT09 / MT12
10%	0.3444 / 0.3770	0.3913 / **0.3809**	**0.4140** / 0.3798
20%	0.3616 / 0.3717	**0.4163** / **0.3792**	0.4129 / 0.3785
30%	0.3787 / 0.3803	0.4095 / **0.3806**	**0.4179** / 0.3746
40%	0.3736 / **0.3813**	0.4046 / 0.3810	**0.4187** / 0.3810

For the submodular method, restricting the maximum phrase length can sometimes be beneficial. This seems to depend very much on the test data set. Table 16 shows that while a maximum phrase length of 10 is optimal for the MT09 data set, a smaller maximum phrase length helps with the MT12 data set. The performance of the different systems on the development set largely mirrors their performance on the test set; it would therefore be possible to optimize the maximum phrase length parameter on the development data.

Overall, we see a distinct gain of the submodular method over the baseline methods. Its superior performance is due to its inherent ability to filter out redundancy among the selected data. Unlike the modular cross-entropy method it avoids over-coverage of already-selected words and phrases.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

4.3 Other Results

***Submodular Feature Selection*:** In addition to the ASR and MT experiments described above several other experiments and techniques have resulted from this project, notably a submodular approach to the problem of feature selection.

Our initial experiments with Fisher kernels showed that the high dimensionality of Fisher score spaces is problematic. Fisher scores spaces are created by taking the derivative of the data log-likelihood with respect to every parameter in all models in the system under consideration; even for a simple phone recognition system with 48 3-state HMMs with 4-64 Gaussians each, this resulted in a score space with hundreds of thousands of dimensions. Not only did this slow down computation it also made the kernel similarity measure less accurate since many dimensions may be noisy or irrelevant. It was necessary to select a subset of features; this type of feature selection can be approached as another submodular selection problem, where the goal is to select a subset of features that expresses all of the information of the original large set but eliminates redundancy among features. Formally, the ground set V is now a set of features rather than samples; A is a subset of features. The facility location and saturated graph cut functions can be applied as before; the similarity measure w now measures the similarity between different features. Here, the mutual information between features was used as w; mutual information is computed from discretized versions of the continuous features.

We applied submodular feature selection to three different problems: (a) improving the Fisher kernel similarity measure for the graphs used in our data subset selection approach; (b) improving Fisher kernel similarity for graph-based semi-supervised phonetic classification; and (c) feature selection for classifying speaker types (four types of neuro-developmental disorders) from acoustic signals.

4.3.1. Results for Data Subset Selection

Initially submodular feature selection was applied to reducing the dimensionality of the Fisher score space used for computing similarity measures for the graph-based submodular functions used for the TIMIT task (described in Section REF). The initial score space for the system consisting of 48 phone HMMs with 16-component diagonal-covariance Gaussian mixtures was 186,577. Experiments were conducted with submodular feature selection, using the facility location function, and modular feature selection for comparison. Modular feature selection was done by computing the mutual information between each feature and the phonetic class label (again from discretized versions of the features) and ranking all features in descending order of the mutual information value. The top N features were then selected, where N was varied between 1,000 and 50,000. The following table shows that the relative improvement in phone accuracy rates of the submodular data selection procedure is greater when submodular feature selection is used in the Fisher kernel rather than modular feature selection.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 17: Relative Improvements in Phone Accuracy on TIMIT for Different Subset Sizes and Different Numbers of Features

(Submod = Submodular Selection, Mod = Modular Selection)

	2.5%	5%	10%	20%	30%
1k - submod	1.99	2.17	2.48	1.48	1.93
1k - mod	-0.20	-0.49	1.48	0.52	1.52
2k - submod	0.11	1.60	2.33	1.46	1.97
2k - mod	0.04	-0.59	0.45	1.24	1.45
5k - submod	0.84	1.15	2.64	1.12	1.37
5k - mod	1.06	1.46	-0.41	-0.30	-0.18
10k - submod	5.21	3.05	3.08	1.16	2.84
10k - mod	2.79	1.29	1.03	0.97	0.92
20k - submod	6.45	3.36	4.34	2.58	1.46
20k - mod	3.77	2.34	3.61	0.79	1.16
50k - submod	4.79	5.27	3.92	2.79	2.71
50k - mod	2.55	3.85	2.48	1.38	2.11

4.3.2. Results for Graph-Based Semi-Supervised Phone Classification

Graph-based semi-supervised learning is a machine learning technique where labeled training data and unlabeled test data are jointly represented in a data graph where nodes represent samples and edges represent the similarity between different samples. A learning algorithm then learns a function that assigns labels to the unlabeled samples, using the data graph as a constraint on the label assignment. For this scenario a similarity measure is required to build the graph. Our work uses the Fisher kernel to build a similarity graph over speech data and then applies a semi-supervised learning algorithm (either measure propagation [26] or label propagation [27]) to infer phonetic class labels. Table 18 shows the effectiveness of this method with and without submodular feature selection. The phone accuracy rates obtained with submodular feature selection are significantly higher than those obtained with modular feature selection; the difference is especially marked for the smallest feature set of 400 features.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

Table 18: Frame-Based Phone Classification Accuracy for Different Numbers of Features

(LP = Label Propagation; MP = Measure Propagation)

# Features	400		800		2k		4k		10k	
	LP	MP	LP	MP	LP	MP	LP	MP	LP	MP
Modular	42.07	42.95	63.83	64.23	62.80	64.09	68.05	68.99	63.48	64.30
Submodular	66.64	67.16	69.43	70.45	68.87	69.25	69.24	69.48	67.04	67.14

4.3.3. Results for Classification of Speaker Types

Finally, we applied submodular feature selection to the Interspeech 2013 Paralinguistics Challenge Task of classifying developmental disorders from speech [28]. The task was to classify utterances described by a large set of acoustic-prosodic features into four speaker types: pervasive developmental disorder (PDD), pervasive developmental disorder – not otherwise specified (PDD-NOS), typical, and dysphasia. In addition to the 4-way classification task, a 2-way classification task asked for the identification of "typical" vs. "non-typical" speakers, thus collapsing the three non-typical classes into one. Our experiments utilized submodular feature selection to condense the original set of 6,373 features into a smaller set. The classifier utilized for this task was a 3-layer multi-layer perceptron (MLP). Table 19 shows the performance with and without submodular feature selection, and the performance of the official Challenge baseline system. On the development set, the system based on submodular feature selection outperforms the official system provided by the Challenge organizers, the MLP with all 6,373 features (baseline), and a system utilizing modular feature selection. On the evaluation set, the submodular system outperforms the official baseline on the 2-way classification task but not on the 4-day task – the likely reason is that the official system was retrained on the combined training and development data. In our case some of this data had to be held out to determine the stopping criterion for training the MLP. The starred systems (baseline and modular systems) were not tested on the evaluation data since the data was not released to sites, and only a limited number of uploads to the official evaluation server were permitted.

Table 19: Unweighted Average Recall for Speaker Type Classification on Interspeech 2013Paralinguistics Autism Sub-challenge Task

	Development		Evaluation	
	2-way	4-way	2-way	4-way
Official	92.8	51.7	90.7	67.1
Baseline	93.7	51.6	N/A*	N/A*
Modular	92.7	54.2	N/A*	N/A*
Submodular	**94.1**	**56.5**	**92.6**	64.4

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

4.4 Publications

The following papers were published as the result of this project. Additional papers are currently in submission.

1. Y. Liu, K. Wei, K. Kirchhoff, Y. Song and J. Bilmes, "Submodular Feature Selection forHigh-Dimensional Acoustic Score Spaces", *Proceedings of ICASSP*, 2013, pp. 7184-7187.

2. K. Wei, Y. Liu, K. Kirchhoff and J. Bilmes, "Using Document Summarization Techniques forSpeech Data Subset Selection", *Proceedings of NAACL*, 2013, pp. 721-726.

3. K. Kirchhoff, Y. Liu and J. Bilmes, "Classification of Developmental Disorders from SpeechSignals using Submodular Feature Selection", *Proceedings of Interspeech*, 2013, pp. 187-190.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

5.0 CONCLUSIONS

We have described the development of a novel approach to data subset selection based on submodular function maximization. This approach consists in defining functions that are monotone submodular and express desirable properties of the speech or language processing system under investigation (e.g., coverage of test data for MT systems). The resulting function is then maximized by a greedy algorithm. Our method was evaluated on both small-scale and large-scale ASR and MT tasks; results demonstrated that submodular data selection leads to better system performance than random baseline selection and modular data selection methods under nearly all conditions, confirming our initial hypothesis made on the basis of the theoretical performance guarantees offered by submodular optimization. The work performed in this project has allowed us to gain better insight into the design of appropriate submodular functions for particular applications in speech and language processing. In particular, submodular function design involves a combination of domain and task-specific insights (knowledge of what properties of a system lead to high performance scores, such as coverage of higher-order n-grams in MT) and experimental investigation of variable parameters in possible functions (e.g., values of weighting factors, or different instantiations of concave functions). Another important aspect that emerged during this work is the computational complexity of computing necessary parameters of certain submodular functions, especially when working on large data sets. Thus, functions requiring pairwise similarity measures may be prohibitively expensive or at least require significant engineering effort in order to be applicable to large data sets. However, alternative functions, such as the feature-based functions developed as part of this work, can often be found and applied successfully.

In addition to data subset selection other applications of submodularity were investigated, including submodular feature selection. On all tasks and under nearly all conditions the submodular selection approach outperformed the comparable baseline methods. We conclude that the submodular framework is a very promising approach not only for the problem of data subset selection but also for other related problems, such as feature selection and summarization.

There are several possible future extensions of the work begun in this project: First, data subset selection can be applied to many other data collections of a different nature, including image or video data, geolocation data, sensor data, etc. Second, the principle of submodularity can be applied to other learning problems that are widely applicable in speech or language processing, such as speech and text summarization, lattice pruning, word clustering, word sense disambiguation, etc. Third, many of the submodular functions presented in Section 3 have not yet been fully explored or evaluated on real-world data sets. Fourth, the methods presented in Section 4 can be extended in various ways, e.g., the functions used for ASR can be modified to include more direct acoustic or prosodic measures; the functions used for MT can be changed to work on bilingual rather than monolingual data. In summury, the submodular framework is an extremely promising approach a number of selection problems in speech and language processing as well as other areas, and exploration of this framework is still in its early stages.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

6.0 REFERENCES

1. Moore, R., Lewis, W., "Intelligent Selection of Language Model Training Data," *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010, pp.

220-224.

2. Axelrod, A., He, X., Gao, J., "Domain Adaptation via Pseudo In-Domain Data Selection," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011, pp. 355-362

3. Mansour, S., Wuebker, J. & Ney, H., "Combining Translation and Language Model Scoring for Domain-Specific Data Filtering," *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, 2011, pp. 222-229.

4. Wu, Y., Zhang, R. & Rudnicky, A., "Data Selection for Speech Recognition," *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2007, pp. 562–565.

5. Edmonds, J. **Combinatorial Structures and their Applications.** Gordon and Breach, 1970.

6. Nemhauser, G.L., Wolsey, L.A. & Fisher, M.L., An Analysis of Approximations for Maximizing Submodular Set Functions---I. *Mathematical Programming*, **14(1**), 1978, pp. 265-294.

7. Wolsey, L.A., "An Snalysis of the Greedy Algorithm for the Submodular Set Covering Problem," *Combinatorica*, **2(4**), 1982, pp. 385-93.

8. Friedland, S. & Gaubert, S., "Submodular Spectral Functions of Principal Submatrices of a Hermitian Matrix, Extensions and Applications," *Linear Algebra and its Applications*, **51**, 2011, pp. 199-208.

9. Cornuejols, G., Fisher, M., Nemhauser, G.L., "On the Uncapacitated Location Problem," *Annals of Discrete Mathematics*, **1**, 1977, pp. 63-177.

10. Bicici, E., Yuret, D., "Instance Selection for Machine Translation using Feature Decay Algorithms," *Proceedings of the Workshop on Machine Translation (WMT)*, 2011, pp. 272-283.

11. Stobbe, P., Krause. A., "Efficient Minimization of Decomposable Submodular Functions," In *Advances in Neural Information Processing Systems (NIPS)*, 2010, pp. 2208-2216.

12. Lin, H., Bilmes, J., "A Class of Submodular Functions for Document Summarization." *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2011, pp. 510-520.

13. Lin, H., Bilmes, J., "Learning Mixtures of Submodular Shells with Application to Document Summarization," In *Proceedings of UAI*, 2012.

14. Guillory, A, Bilmes, J., "Simultaneous Learning and Covering with Adversarial Noise," In *Proceedings of the International Conference on Machine Learning (ICML),* 2011, pp. 369-376.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

15. Jaakkola, T. S., Haussler, D. ,(1999a), "Exploiting Generative Models in Discriminative Classifiers," in Kearns, M.S.. Solla, A., Cohn, D.A. (eds.), *Advances in Neural Information Processing Systems (NIPS) 11*, Bradford Books, The MIT Press, Cambridge, MA, pp. 487–493.

16. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C., "Text Classification Using String Kernels, " *Journal of Machine Learning Research*, **2**, 2002, pp. 419-444.

17. Cortes, C., Mohri, M., "Rational Kernels: Theory and Algorithms," *Journal of Machine Learning Research (JMLR)*, **5**, 2004, pp. 1035-1062.

18. Lee, K., Hon, H. (1989)., "Speaker-Independent Phone Recognition using Hidden Markov Models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **37 (11**), 1989, pp. 1642-1648.

19. Nagendra, K., Andreou, G., "*Investigation of Silicon Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition*", Diss. Johns Hopkins University, 1997.

20. Povey, D., Kingsbury, B., Mangu, L., Saon, G., Soltau, H., Zweig, G., "fMPE: Discriminatively Trained Features for Speech Recognition", in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2005, pp. 961–964.

21. Povey, D., Woodland, P.C., "Minimum Phone Error and I-smoothing for Improved Discriminative Training," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP*), 2002, pp. 961-964.

22. Stolcke, A. et al., "The SRI March 2000 Hub-5 Conversational Speech Transcription System," *Proceedings of the NIST Speech Transcription Workshop*, 2000.

23. Habash, N., Rambow, O., Roth, R., "MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization," *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, 2009, pp. 102-109.

24. Hoang, H., Koehn, P., "Design of the Moses Decoder for Statistical Machine Translation," *Proceedings of the ACL Workshop on Software Engineering, Testing, and Quality Assurance for NLP*, 2008, pp. 58-65.

25. Galley, M., Manning, C., "A Simple and Effective Hierarchical Phrase Reordering Model", *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP*), 2008, pp. 847-855.

26. Subramanya, A., Bilmes, J., "Semi-supervised Learning with Measure Propagation," Technical Report, *UW-EE-TR-2010-0004*, University of Washington, 2010.

27. Zhu, S., Ghahramani, Z., "Learning from Labeled and Unlabeled Data with Label Propagation," Technical Report, *CMU-CALD-02*, Carnegie Mellon University, 2002.

28. Schuller, B. et al., "The INTERSPEECH 2013 Computational Paralinguistics Challenge: Social Signals, Conflict, Emotion, Autism," *Proceedings of Interspeech*, 2013.

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

ASR	Automatic Speech Recognition
BLEU	Bilingual Evaluation Understudy *(evaluation measure for machine translation*)
BOLT	Broad Operational Language Translation *(current DARPA research project)*
CPU	Central Processing Unit
DARPA	Defense Advanced Research Projects Agency
DECIPHER	*(name of a speech recognition system)*
DSSS	Data Subset Selection
fMPE	Feature Minimum Phone Error
FST	Finite-State Transducer
GALE	Global Autonomous Language Exploitation *(former DARPA research project)*
GIZA++	*(software for word alignment)*
GMM	Gaussian Mixture Model
HLDA	Heteroscedastic Linear Discriminant Analysis
HMM	Hidden Markov Model
IARPA	Intelligence Advanced Research Projects Agency
LM	Language Model
LP	Label Propagation
MADA	*(software package for Arabic morphological processing)*
MERT	Minimum Error Rate Training
MFCC	Mel-Frequency Cepstral Coefficient
MLLR	Maximum Likelihood Linear Regression
MLP	Multi-Layer Perceptron
MP	Measure Propagation
MPE	Minimum Phone Error
MSA	Modern Standard Arabic
MT	Machine Translation
NIST	National Institute for Standards and Technology
NLP	Natural Language Processing
PER	Position-Independent Word Error Rate
PDD	Pervasive Developmental Disorder
PDD-NOS	Pervasive Developmental Disorder – Not Otherwise Specified
SCCM	Sum of Concave over Modular Functions
SRI	Stanford Research Institute

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013

TF-IDF	Term Frequency – Inverse Document Frequency
TIMIT	*(name of a speech corpus)*
URL	Uniform Resource Locator
UTF	Unicode Transformation Format

Disribution A. Approved for public release; distribution unlimited.
88ABW-2013-5110; Cleared 04 December 2013